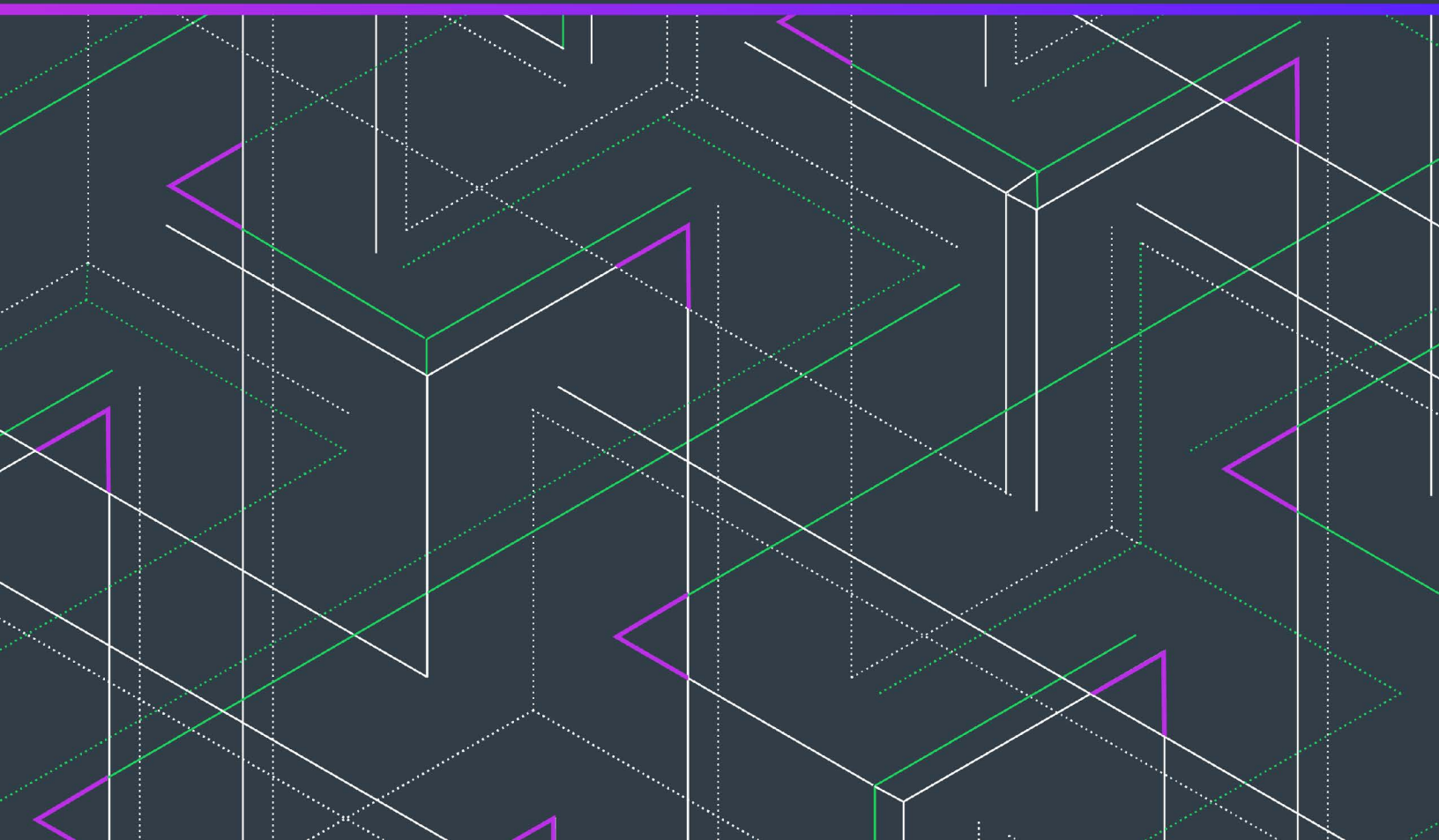
revenera™

InstallShield 2026 Express Edition

User Guide



Legal Information

Book Name: InstallShield 2026 Express Edition User Guide

Part Number: ISE-3200-UG00

Product Release Date: September 2026

Copyright Notice

Copyright © 2026 Flexera. All Rights Reserved.

This publication contains proprietary and confidential information and creative works owned by Flexera and its licensors, if any. Any use, copying, publication, distribution, display, modification, or transmission of such publication in whole or in part in any form or by any means without the prior express written permission of Flexera is strictly prohibited. Except where expressly provided by Flexera in writing, possession of this publication shall not be construed to confer any license or rights under any Flexera intellectual property rights, whether by estoppel, implication, or otherwise.

All copies of the technology and related information, if allowed by Flexera, must display this notice of copyright and ownership in full.

Intellectual Property

For a list of trademarks and patents that are owned by Flexera, see https://www.flexera.com/producer/company/about/intellectual-property/ . All other brand and product names mentioned in Flexera products, product documentation, and marketing materials are the trademarks and registered trademarks of their respective owners.

Restricted Rights Legend

The Software is commercial computer software. If the user or licensee of the Software is an agency, department, or other entity of the United States Government, the use, duplication, reproduction, release, modification, disclosure, or transfer of the Software, or any related documentation of any kind, including technical data and manuals, is restricted by a license agreement or by the terms of this Agreement in accordance with Federal Acquisition Regulation 12.212 for civilian purposes and Defense Federal Acquisition Regulation Supplement 227.7202 for military purposes. The Software was developed fully at private expense. All other use is prohibited.

Contents

1 InstallShield 2026 Express Edition Help Library 21

What Was New in Earlier Versions of InstallShield Express Edition 22

What's New in InstallShield 2025 Express Edition 23
What's New in InstallShield 2023 Express Edition 24
What's New in InstallShield 2022 Express Edition 27
What's New in InstallShield 2021 Express Edition 27
What's New in InstallShield 2020 Express Edition 28
What's New in InstallShield 2019 Express Edition 29
What's New in InstallShield 2018 Express Edition SP1 32
What's New in InstallShield 2018 Express Edition 32
What's New in InstallShield 2016 SP2 Express Edition 36
What's New in InstallShield 2016 SP1 Express Edition 37
What's New in InstallShield 2016 Express Edition 37
What's New in InstallShield 2015 SP1 Express Edition 40
What's New in InstallShield 2015 Express Edition 41
What's New in InstallShield 2014 SP1 Express Edition 44
What's New in InstallShield 2014 Express Edition 44
What's New in InstallShield 2013 SP1 Express Edition 46
What's New in InstallShield 2013 Express Edition 50
What's New in InstallShield 2012 Spring SP1 Express Edition 52
What's New in InstallShield 2012 Spring Express Edition 52
What's New in InstallShield 2012 SP1 Express Edition 54
What's New in InstallShield 2012 Express Edition 54
What's New in InstallShield 2011 Express Edition 56
What's New in InstallShield 2010 Express Edition 63
What's New in InstallShield 2010 Express Edition SP1 64
What's New in InstallShield 2010 Express Edition 65
What's New in InstallShield 2009 Express Edition 69
What's New in InstallShield 2008 Express Edition 74
What's New in InstallShield 12 Express Edition 81

Target System Requirements **83**
Challenges of Supporting Both 32-Bit and 64-Bit Target Operating Systems 85
Launching InstallShield with vs. Without Administrative Privileges **88**
Developing and Building Installations on 32-Bit vs. 64-Bit Systems **89**
Using Help **91**
Contacting Us **93**

2 Getting Started **95**
Installation Fundamentals **96**
Overview of Installations **97**
Application Lifecycle **98**
Starting InstallShield **99**
InstallShield Start Page 99
Working with Projects **101**
Project Types 101
Express Projects 101
QuickPatch Projects 101
Using Projects 102
Creating New Projects 102
Opening Projects 102
Opening Projects Created with Earlier Versions of InstallShield 103
Saving Projects 103
Saving a Project with a New Name and Location *103*
Changing the Default Project Location 104
GUIDs 104
Sample Files **105**
Project Assistant **105**
Using the Project Assistant 106
Navigating in the Project Assistant 106
Opening the Installation Designer 106
Hiding the Project Assistant 107
Application Information Page 107
Add or Remove Programs in the Control Panel 107
Company Name and Product Name in Your Installation 107
Installation Requirements Page 107
Specifying Operating System Requirements in the Project Assistant 108
When Does the Installation Check for Requirements? 108
Modifying the Run-Time Message for Software Requirements 108
Creating Custom Installation Requirements 108
Installation Architecture Page 109
Adding Features in the Project Assistant 109
Determining Whether to Create a Multiple-Feature Installation 109
Creating Installations with Multiple Features 109
Default Features 110
Defining Feature Hierarchy 110

Application Files Page . . . 110
Adding Files to Features in the Project Assistant . . . 111
Removing Files from Features in the Project Assistant . . . 111
Adding Files to a Fixed Folder Location . . . 111
Viewing Additional Predefined Folders . . . 111
Application Shortcuts Page . . . 112
File Extensions . . . 112
Creating Shortcuts to Files That Are Not Included in the Installation . . . 112
Modifying a Default Shortcut in the Project Assistant . . . 113
Associating a Shortcut's Target with a File Extension in the Project Assistant . . . 113
Application Registry Page . . . 113
Updating the Registry . . . 114
Configuring Registry Data in the Project Assistant . . . 114
Modifying Registry Data Values in the Project Assistant . . . 114
Associating Registry Data with Features . . . 114
Using Variable Data Types in Registry Data . . . 115
Application Paths . . . 115
Installation Interview Page . . . 115
Specifying Dialogs for Your Installation in the Project Assistant . . . 116
Allowing End Users to Modify the Installation Location . . . 116
License Agreements . . . 116
Creating Selectively Installable Installations . . . 117
Build Installation Page . . . 117
Building Your Installation from the Project Assistant . . . 117
After Completing the Project Assistant: Next Steps . . . 118

Working with the InstallShield Interface . . . 118

Displaying the View List . . . 118
Opening Views in the InstallShield User Interface . . . 119
Working with the Group Box Area in Various Views . . . 119
Showing or Hiding Toolbars . . . 121
Adding Buttons and Menus to a Toolbar . . . 121
Removing Buttons and Menus from a Toolbar . . . 122
Creating Custom Toolbars . . . 122
Docking or Undocking the Output Window . . . 122

Configuring Advanced Settings for InstallShield . . . 123

Changing the Timestamp Server for Digital Signatures . . . 123
Configuring the Compression Level for Files that Are Streamed into Setup.exe . . . 125
Configuring the Maximum Size for .cab Files . . . 126
Configuring Platform Architecture for Digital Signing . . . 127

Upgrading from Earlier Versions of InstallShield . . . 128

Upgrading Projects from InstallShield 2014 Express Edition or Earlier . . . 129
Upgrading Projects from InstallShield 2013 Express Edition or Earlier . . . 130
Upgrading Projects from InstallShield 2012 Spring Express Edition or Earlier . . . 130
Upgrading Projects from InstallShield 2012 Express Edition or Earlier . . . 131
Upgrading Projects from InstallShield 2011 Express Edition or Earlier . . . 132
Upgrading Projects from InstallShield 2010 Express Edition or Earlier . . . 133

Upgrading Projects from InstallShield 2009 Express Edition or Earlier . . . 135
Upgrading Projects from InstallShield 2008 Express Edition or Earlier . . . 138
Upgrading Projects from InstallShield 12 Express Edition or Earlier . . . 141
Upgrading Projects from InstallShield Express 2.x . . . 143
Upgrading to the Premier or Professional Editions of InstallShield . . . 146
Converting or Importing Visual Studio Projects into InstallShield Projects . . . 150
Obtaining Updates for InstallShield . . . 154
Run-Time Language Support in InstallShield . . . 154
Supported Run-Time Languages . . . 155
Supported Application Programming Languages . . . 157

3 Tutorials . . . 159
Basic Tutorial . . . 161
Create a New Project . . . 161
Organize Your Setup . . . 161
Specify Application Data . . . 163
Configure the Target System . . . 163
Customize the Setup Appearance . . . 164
Define Setup Requirements and Actions . . . 165
Prepare for Release . . . 165
Summary . . . 166

4 Creating Installations . . . 167
Before You Begin . . . 168
Introduction to Windows Installer . . . 169
Minimizing the Number of User Account Control Prompts During Installation . . . 169
Requirements for the Windows Logo Program . . . 173
Preventing the Current Installation from Overwriting a Future Major Version of the Same Product . . . 173
Preparing Installations for Non-Administrator Patches . . . 174
Specifying Installation Information . . . 175
Configuring General Project Settings . . . 175
Setting the Product Code . . . 175
Specifying a Product Name . . . 175
Specifying the Product Version . . . 176
Setting the Upgrade Code . . . 176
Configuring Add or Remove Programs Information . . . 176
Entering Summary Information Stream Data . . . 177
Setting the Default Product Destination Folder (INSTALLDIR) . . . 177
INSTALLDIR and the Registry . . . 178
Setting INSTALLDIR from the Registry . . . 178
Securing Files, Folders, and Registry Keys in a Locked-Down Environment . . . 178
Selecting the Locked-Down Permissions Type for a Project . . . 180
Specifying Whether Windows Installer Installations Should Be Logged . . . 181
Including a Software Identification Tag for Your Product . . . 182
Organizing Files for Your Installation . . . 185

Designing Installations . . . 185
Separating Applications into Features . . . 185
Defining Features . . . 186
Creating Features . . . 186
Creating Subfeatures . . . 187
Configuring Feature Settings . . . 187
Setting Feature Conditions . . . 187
Displaying Features to End Users . . . 188
Requiring Features to Be Installed . . . 189
Advertising Features . . . 189
Setting a Feature's Remote Installation Setting . . . 190
Changing the Feature Order in the Custom Setup Dialog . . . 190
Working with Setup Types . . . 190
Specifying Features Included in Setup Types . . . 191
Renaming Setup Types . . . 191
Specifying Setup Types . . . 191
Including Only One Setup Type in an Installation . . . 192
Accessing the Setup Type at Run Time . . . 192
Including Files and Folders . . . 192
Adding Files and Folders to a Project . . . 193
Dragging and Dropping Files Using the Context Menu . . . 194
Adding Files from Your 64-Bit Source Machine's 64-Bit System32 Folder . . . 196
Specifying Hard-Coded Destination Directories . . . 197
Removing Files and Folders from Target Systems . . . 197
Tips for Managing Files and Folders in Your Project . . . 198
Specifying Target System Requirements for Individual Files . . . 199
Dynamic File Linking . . . 199
Limitations of Dynamic File Linking . . . *200*
Determining the Appropriate Component Creation Method for Dynamically Linked Files . . . *201*
Adding Files Dynamically . . . *202*
Overwriting Files on the Target Machine . . . 203
Displaying Predefined Folders in the Files View . . . 204
Finding Files and Folders in Your Project . . . 204
Configuring Permissions for Files and Folders . . . 204
Including Redistributables in Your Installation . . . 205
Shipping Redistributable Files . . . 206
Managing the Redistributables Gallery . . . 208
Downloading Redistributables to Your Computer . . . *210*
Adding InstallShield Prerequisites to the Redistributables Gallery . . . *210*
Removing InstallShield Prerequisites from the Redistributables Gallery . . . *211*
Browsing for Merge Modules . . . *211*
What Happens When You Browse for a Merge Module . . . *211*
Adding Merge Modules to the Redistributables Gallery . . . *212*
Removing Merge Modules from the Redistributables Gallery . . . *212*
Incorporating InstallShield Prerequisites, Merge Modules, and Objects in Projects . . . 213
Adding InstallShield Prerequisites, Merge Modules, and Objects to Projects . . . *213*

Removing InstallShield Prerequisites, Merge Modules, or Objects from a Project 214
Determining the Files in InstallShield Prerequisites, Merge Modules, and Objects 214
Working with InstallShield Prerequisites that Are Included in Projects 214
Setup Prerequisites vs. Feature Prerequisites 215
Associating an InstallShield Prerequisite with a Feature in a Project 216
Disassociating an InstallShield Prerequisite from a Feature in a Project 217
Specifying the Installation Order of InstallShield Prerequisites 217
Configuring a Release that Includes InstallShield Prerequisites 218
Specifying the Directories that Contain InstallShield Prerequisites 218
Specifying a Run-Time Location for a Specific InstallShield Prerequisite 219
Building a Release that Includes InstallShield Prerequisites 219
Run-Time Behavior for an Installation that Includes InstallShield Prerequisites 220
Uninstalling an Application Whose Installation Included InstallShield Prerequisites 223
Working with Merge Modules and Objects that Are Included in Installation Projects 223
Specifying the Directories that Contain Merge Modules 224
Modifying the Object and Merge Module Configurations 225
Merge Module Exclusions and Dependencies 225
Overriding a Merge Module's Destination 225
Troubleshooting Merge Module Issues 226
Adding Windows Installer Redistributables to Projects 226
Including Microsoft Windows Installer Prerequisites 227
Adding .NET Framework Redistributables to Projects 227
Including the Microsoft .NET Framework and Microsoft .NET Framework Language Pack Prerequisites . . . 228
Including the DirectX 9.0 Object 228
Identifying Application Dependencies 230
Static Scanning 230
Dynamic Scanning 230
Reviewing Dependency Scanner Results 230
Filtering Files in Dependency Scanners 231
Registering COM Servers 233
Filtering Registry Changes for COM Extraction 233
Configuring the Target System 237
Creating Shortcuts and Program Folders 237
Types of Shortcuts 237
Creating Shortcuts 237
Specifying the Icon for a Shortcut 238
Basing a Shortcut on the Source Media 239
Specifying a Keyboard Shortcut for Accessing a Shortcut 239
Renaming Shortcuts 240
Creating an Uninstallation Shortcut 240
Configuring the Appearance of a Desktop App's Tile on the Start Screen 241
Editing the Registry 241
Filtering Registry Entries by Feature 243
Creating a Registry Key 243
Dragging and Dropping Registry Entries to Create Registry Keys 244
Importing Registry Data from a .reg File 245

Removing Registry Keys . . . 246
Creating Registry Values . . . 246
Modifying Registry Value Data . . . 247
Removing Registry Values . . . 247
Entering Multiple Registry String Values into a Single String . . . 248
Referencing an Environment Variable in a Registry Entry . . . 248
Configuring Permissions for Registry Keys . . . 249
Specifying a Primary Key for the Registry Table . . . 249
Registry Flags . . . 250
Setting Install/Uninstall Behavior for Registry Keys . . . 251
Handling Registry Entries for a Per-User Installation . . . 251
Refreshing the Registry View . . . 251
Associating a File Extension with an Application File . . . 251
Creating a File Extension Association . . . 252
Changing .ini File Data . . . 252
Adding an .ini File . . . 253
Importing an Existing .ini File . . . 253
Specifying a Section in an .ini File . . . 254
Specifying a Keyword and its Value in an .ini File . . . 254
Configuring ODBC Resources . . . 254
Including an ODBC Resource . . . 255
Including Additional ODBC Resources . . . 255
Associating an ODBC Resource with a Feature . . . 255
Setting the Attributes for an ODBC Resource . . . 256
Adding a New Attribute to an ODBC Resource . . . 256
Removing an Attribute from an ODBC Resource . . . 257
Using Environment Variables . . . 257
Setting Environment Variables . . . 257
Installing and Configuring Windows Services . . . 257
Per-User vs. Per-Machine Installations . . . 258
Customizing Installation Behavior . . . 261
Using Custom Actions . . . 261
Windows Installer DLL Custom Actions . . . 263
Adding an MSI DLL Custom Action to Your Project . . . 264
Configuring an MSI DLL Custom Action's Settings . . . 264
DLL Custom Actions . . . 265
Classic DLL Custom Action Function Prototype . . . 265
New DLL Custom Action Function Prototype . . . 266
Adding a DLL Custom Action to Your Project . . . 267
Configuring a DLL Custom Action's Settings . . . 267
Executable File Custom Actions . . . 268
Adding an .exe Custom Action to Your Project . . . 268
Configuring an .exe Custom Action's Settings . . . 268
VBScript and JScript Custom Actions . . . 269
VBScript Custom Action Example . . . 269
Adding a VBScript or JScript Custom Action to Your Project . . . 269

Configuring a VBScript or JScript Custom Action's Settings *270*
Action Execution Options 270
Changing When Custom Actions Are Launched 272
Using a Custom Action for Serial Number Validation 272
Custom Action Gallery 273
Checking if a Product Is Installed 273
Using Setup Files 274
Adding Setup Files 275
Adding a License File 275
Accessing a Setup File During Installation 276
Sorting Support Files 276
Disk1 Files 276
Adding Disk1 Files *276*
Adding Disk1 Folders *277*
Removing Disk1 Files and Folders *277*
Removing Setup Files 277
Configuring Servers 279
Managing COM+ Applications and Components 279
Adding COM+ Applications 279
Removing COM+ Applications 280
Managing Internet Information Services 280
Version-Specific Information for IIS Support in InstallShield 280
Run-Time Requirements for IIS Support 281
Specifying Whether a Web Server Should Allow the CMD Command to Be Used for SSI #exec Directives 282
Creating a Web Site and Adding an Application or a Virtual Directory 283
Creating a Nested Virtual Directory *284*
Configuring the TCP Port and Site Numbers 285
Specifying the IIS Host Header Name for a Web Site 287
Specifying the SSL Certificate for a Web Site 288
Adding Files to an IIS Virtual Directory 288
Removing Applications and Virtual Directories from the Internet Information Services View 289
Feature Associations for IIS Support 289
Uninstalling Web Sites, Applications, and Virtual Directories 289
Setting the ASP.NET Version for a Web Site or Application 289
Defining Application Mappings for a Web Site, Application or Virtual Directory 290
Specifying Timeout Parameters for a Web Site or Virtual Directory 290
Configuring Additional IIS Virtual Directory Settings 291
Configuring Custom Error Messages for a Web Site, Application, or Virtual Directory 291
Deploying Web Services on a Target Machine 292
Stopping IIS Functionality Through a Custom Action 292
Enabling Forms Authentication on Web Applications 293
Adding IISROOTFOLDER Support 293
IIS_WEBSITE_NAME Property 293
IIS_PORT_NUMBER Property 294
Defining the End-User Interface 295
Working with Dialogs 295

Adding a Dialog to an Installation . . . 295
Dialog Themes . . . 295
Selecting or Changing a Dialog Theme . . . *295*
Available Themes . . . *296*
Bitmap Images in Dialogs . . . 296
Changing the Splash Bitmap in the Splash Bitmap Dialog . . . *299*
Changing the Bitmap Image for an End-User Dialog . . . *300*
Changing the Banner Bitmap for an End-User Dialog . . . *300*
Changing the Global Dialog Image for End-User Dialogs . . . *301*
Changing the Global Dialog Banner for End-User Dialogs . . . *301*
Minimizing Reboots on Windows Vista and Later Systems . . . 302
Custom Setup Dialog Options . . . 302
Displaying a License Agreement During the Run Time . . . 303
Removing a Dialog from an Installation . . . 303
Editing Run-Time Text and Message Strings . . . 303
Editing Run-Time Strings . . . 304
Adding Comments to Text and Message Strings . . . 304
Changing the Font for Text and Message Strings . . . 304
Exporting Strings . . . 305
Importing String Tables . . . 305
Displaying Billboards . . . 305
Billboard File Types . . . 305
Types of Billboards . . . 306
Specifying Which Type of Billboard to Use in an Installation . . . *308*
Adding an Adobe Flash Application File Billboard . . . 308
Adding Image Billboards . . . 309
Configuring Billboard Settings . . . 309
Previewing Billboards Without Building and Launching a Release . . . 310
Setting the Billboard Order . . . 310
Run-Time Behavior of an Installation that Includes Billboards . . . 311
Removing a Billboard . . . 311
Preparing Installations for Maintenance and Uninstallation . . . 313
Removing Registry Data Created by Your Product . . . 313
Building, Testing, and Distributing Installations . . . 315
Configuring and Building Releases . . . 315
Building a Release . . . 315
Creating a Setup Launcher . . . 316
Customizing File Properties for the Setup Launcher . . . *317*
Building 64-Bit Setup Launcher . . . 320
Canceling Builds . . . 321
Changing the Product Version During a Build . . . 321
Build Release Location . . . 322
Build Logs and Reports . . . 322
Quick Builds . . . 322
Performing Quick Builds . . . *323*
Command-Line Builds . . . 323

Building from the Command Line . . . 323
Passing Command-Line Build Parameters in an .ini File . . . 323
Microsoft Build Engine (MSBuild) . . . 326
Using MSBuild to Build a Release from the Command Line . . . 332
Creating a Single Self-Extracting Installation File . . . 332
Specifying the Required Execution Level for Your Setup Launcher on Windows Vista and Later Platforms . . . 333
Preparing Your Installation for Internet Distribution . . . 334
Creating a Web-Deployable Release . . . 335
Digital Signing and Security . . . 335
Digitally Signing a Release and Its Files at Build Time . . . 337
Specifying the Run-Time Location for InstallShield Prerequisites at the Release Level . . . 338
Media Compression . . . 339
Floppy Disk Distributions . . . 339
Building a Release for Floppy Disk Distribution . . . 340
Setting Volume Labels . . . 340
Creating a Windows Installer Package (.msi) as Output . . . 341
Autorun . . . 341
Enabling Autorun for Your CD or DVD . . . 341
Testing and Running Installations . . . 342
Testing and Running Installations . . . 342
Running an Installation in Silent Mode . . . 342
Conditional Installations . . . 342
Distributing Installations . . . 343
Distributing Releases to a Folder or FTP Site Automatically . . . 343

5 Updating Applications . . . 345

Upgrades Overview . . . 345
Major Upgrades . . . 345
Minor Upgrades . . . 346
Small Updates . . . 347
Patching . . . 347
QuickPatch Projects . . . 347
Determining the Best Upgrade Solution . . . 348
Packaging Options for Upgrades . . . 349
Working with Upgrades and QuickPatch Projects . . . 352
Understanding File Overwrite Rules . . . 352
Updating the Package Code, the Product Version, and the Product Code . . . 353
Creating Full-Installation Upgrades . . . 353
Preventing the Removal of Assemblies from the Global Assembly Cache During Upgrades . . . 354
Adding an Upgrade Item . . . 355
Configuring Upgrade Properties . . . 356
Removing Upgrade Items . . . 356
Patching Considerations . . . 356
Patch Sequencing . . . 357
Patch Uninstallation . . . 358
Non-Administrator Patches . . . 359

Creating a QuickPatch Project and Applying QuickPatch Packages . . . 360
Creating a QuickPatch Project for an Existing QuickPatch . . . 360
Specifying Whether to Streamline the QuickPatch Package . . . 361
Specifying Target Releases for Patching . . . 362
Specifying Custom Actions for the QuickPatch to Execute . . . 362
Adding Files to a QuickPatch . . . 362
Specifying Identification Information for a QuickPatch . . . 363
Enabling the Uninstallation of a QuickPatch . . . 363
Sequencing QuickPatch Packages . . . 364
Signing a QuickPatch Package . . . 364
Password-Protecting a QuickPatch Package . . . 365
Specifying Whether to Build an Update.exe Update Launcher for a QuickPatch Package . . . 365
Customizing File Properties for the Update Launcher . . . *366*
Deleting Files from the Files To Patch Explorer . . . 367
Modifying and Removing Installed Files with a QuickPatch . . . 367
Adding, Modifying, and Deleting Registry Data with a QuickPatch . . . 367
Patching Assemblies in the Global Assembly Cache . . . 368
Applying a QuickPatch . . . 368
Notifying End Users about Upgrades Using FlexNet Connect . . . 369

6 Additional Installation Options . . . 371

Specifying Target System Requirements . . . 371
Specifying Operating System Requirements for Your Product . . . 371
Specifying Processor Requirements for Your Product . . . 372
Specifying RAM Requirements for Your Product . . . 372
Specifying Screen Resolution Requirements for Your Product . . . 372
Specifying Color Depth Requirements . . . 373
Specifying Software Requirements for Your Product . . . 373
Working with Windows Installer Properties . . . 374
Windows Installer Property Reference . . . 376
Specifying that a Public Property Should Be a Restricted Public Property . . . 391
Implementing Serial Number Validation at Run Time . . . 392
Accessing the Serial Number During and After Installation . . . 393

7 Building InstallShield Project in DevOps . . . 395

Azure DevOps Build Extension . . . 395
InstallShield StandAlone Build with Docker . . . 396

8 Integrating InstallShield with External Applications . . . 397

Integrating with Microsoft Visual Studio . . . 397
Creating InstallShield Projects in Microsoft Visual Studio . . . 398
Opening InstallShield Projects in Microsoft Visual Studio . . . 398
Using the VSSolutionFolder Path Variable with Visual Studio Solutions . . . 399
Adding References to Visual Studio Solutions . . . 399
Adding InstallShield Toolbars or Commands to the Visual Studio Toolbar . . . 399

Building Releases in Microsoft Visual Studio . . . 400
Adding .NET Assemblies to a Project . . . 401
Adding Project Output from Web Services or a Web Application . . . 401
Adding .NET Framework Support to an Installation Project . . . 402
Integrating with Microsoft Visual Studio Team Foundation Server . . . 402
Adding InstallShield Projects to Team Explorer . . . 403

9 Reference . . . 405

Menu, Toolbar, and Window Reference . . . 407
Menus . . . 407
File Menu . . . 407
Edit Menu . . . 408
View Menu . . . 408
Go Menu . . . 409
Project Menu . . . 410
Build Menu . . . 411
Tools Menu . . . 411
Help Menu . . . 412
Toolbars . . . 412
Standard Toolbar . . . 413
Output Window . . . 414
Dialog Box Reference . . . 415
.NET 1.1/2.0 Core Language Dialog Box . . . 416
.NET 1.1/2.0 Language Packs Dialog Box . . . 416
Add MIME Type Dialog Box . . . 417
Application Extension Mapping Dialog Box . . . 417
Application Mappings Dialog Box . . . 418
Browse for a Destination File Dialog Box . . . 419
Browse for Directory/Set INSTALLDIR Dialog Box . . . 419
Browse for File Dialog Box . . . 420
Browse for Shortcut Target Dialog Box . . . 421
Browse for Tile Target Dialog Box . . . 422
Certificate Selection Dialog Box . . . 422
Condition Builder Dialog Box . . . 424
Content Source Path Dialog Box . . . 425
Custom Errors Dialog Box . . . 425
Dependencies Dialog Box . . . 425
Dialog Images Dialog Box . . . 426
Digitally Sign Setup Dialog Box . . . 426
Edit Registry Data Dialog Box . . . 428
Error Mapping Properties Dialog Box . . . 429
File Details Dialog Box . . . 429
File Properties Dialog Box . . . 430
General Tab . . . 430
COM & .NET Settings Tab . . . 432
Advanced Tab . . . 433

File Removal Properties Dialog Box . . . 435
Folder Properties Dialog Box . . . 436
General Tab . . . 436
File Linking Tab . . . 437
InstallShield Prerequisites Properties Dialog Box . . . 439
Logging Options for Windows Installer 4.0 and Later Dialog Box . . . 440
Merge Module Configurable Values Dialog Box . . . 441
Merge Module Properties Dialog Box . . . 441
MIME Types Dialog Box . . . 442
MSI Value Dialog Box . . . 442
Multi-Line String Value Dialog Box . . . 442
New Project Dialog Box . . . 443
Options Dialog Box . . . 444
General Tab . . . 444
File Locations Tab . . . 445
Preferences Tab . . . 445
Merge Module Options Tab . . . 446
.NET Tab . . . 447
Files View Tab . . . 447
File Extensions Tab . . . 448
Prerequisites Tab . . . 449
Outputs Dialog Box . . . 450
Permissions Dialog Boxes for Files and Directories . . . 450
Permissions Dialog Boxes for Registry Keys . . . 452
Select Icon Dialog Box . . . 454
Set INSTALLDIR Dialog Box/Set DATABASEDIR Dialog Box . . . 455
Settings Dialog Box . . . 456
MSI Log File Tab . . . 456
System Hardware Requirements Dialog Box . . . 457
Update Merge Module Search Path Dialog Box . . . 457
Upgrade Express Project Name Dialog Box . . . 457
Wizard Reference . . . 459
Create New QuickPatch Wizard . . . 459
Welcome Panel . . . 459
Project Name Panel . . . 460
Release to Patch Panel . . . 460
Final Panel . . . 460
DirectX Object Wizard . . . 460
Welcome Panel . . . 461
Object Settings Panel . . . 461
Summary Panel . . . 462
Dynamic Scanning Wizard . . . 462
Welcome Panel . . . 462
Filter Files Panel . . . 463
Specify the Executable Panel . . . 463
Specify Application File Panel . . . 463

Launch the Application . . . 464
Your Application Is Running Panel . . . 464
File Selection Panel . . . 464
Scan Results Panel . . . 464
Completing the Dynamic Scanning Wizard Panel . . . 465
Export String Table Wizard . . . 465
Welcome Panel . . . 465
File Name Panel . . . 465
Wizard Complete Panel . . . 465
Import REG File Wizard . . . 465
Welcome Panel . . . 466
Import Registry File Panel . . . 466
Import Conflict Options Panel . . . 467
Import Progress Panel . . . 467
Import String Table Wizard . . . 467
Welcome Panel . . . 467
File Name Panel . . . 468
Wizard Complete Panel . . . 468
Redistributable Downloader Wizard . . . 468
Static Scanning Wizard . . . 468
Welcome Panel . . . 469
Filter Files Panel . . . 469
Scanning Progress Panel . . . 469
File Selection Panel . . . 470
Scan Results Panel . . . 470
Completing the Static Scanning Wizard Panel . . . 470
System Search Wizard . . . 471
Welcome Panel . . . 471
What do you want to find? Panel . . . 471
How do you want to look for it? Panel (Defining Your System Search Method) . . . 472
Specify the file details Panel (File Search Options) . . . *473*
How do you want to look for it? Panel (Folder Search Options) . . . *473*
How do you want to look for it? Panel (Specific Folder Options) . . . *474*
How do you want to look for it? Panel (Registry Search Options) . . . *474*
How do you want to look for it? Panel (.ini File Search Options) . . . *475*
What do you want to do with the value? Panel . . . 475
Visual Studio .NET Wizard for Visual Basic .NET, Visual C++ .NET, and C# .NET . . . 475
Welcome Panel . . . 476
Solution Panel . . . 476
Visual Studio Deployment Project Import Wizard . . . 476
Welcome Panel . . . 477
Project File Panel . . . 477
Options Panel . . . 477
Summary Panel . . . 481
Web Deployment Wizard . . . 482
Welcome Panel . . . 482

Link Type Panel . . . 482
Windows Installer Engine Options Panel . . . 482
Windows Installer Location Panel . . . 483
Advanced Settings Panel . . . 483
Digital Signature and Security for Targeting Internet Explorer Panel . . . 484
Summary Panel . . . 486
View Reference . . . 487
Organize Your Setup View . . . 487
General Information View . . . 488
General Information Settings . . . 488
Features View . . . 502
Feature Settings . . . 503
Setup Types View . . . 505
Upgrade Paths View . . . 505
Upgrade Path Settings . . . 506
Update Notifications View . . . 510
Specify Application Data View . . . 510
Files View . . . 511
Destination Folders . . . 512
Files and Features View . . . 515
Redistributables View . . . 515
Dependencies View . . . 518
Configure the Target System View . . . 519
Shortcuts/Folders View . . . 520
Shortcut Settings . . . 521
Folder Settings . . . 527
Tile Configuration Settings . . . 527
Registry View . . . 529
ODBC Resources View . . . 529
ODBC Resource Settings . . . 530
INI File Changes View . . . 531
Settings for .ini Files . . . 532
Settings for .ini File Keywords . . . 533
File Extensions View . . . 533
File Extension Settings . . . 534
Environment Variables View . . . 535
Environment Variable Settings . . . 536
Internet Information Services View . . . 538
Web Site Settings . . . 538
Application and Virtual Directory Settings . . . 547
Component Services View . . . 553
Installation Tab . . . 554
Services View . . . 555
Services View Settings . . . 555
Customize the Setup Appearance View . . . 558
Dialogs View . . . 558

Billboards View . . . 559
Billboard Settings . . . *560*
Settings for Adobe Flash Application File Billboards and Image Billboards . . . *561*
Text and Messages View . . . 564
Define Setup Requirements and Actions View . . . 564
Requirements View . . . 564
Custom Actions View . . . 565
MSI DLL Custom Action Settings . . . *566*
DLL Custom Action Settings . . . *568*
Executable File Custom Action Settings . . . *570*
VBScript Custom Action Settings . . . *573*
JScript Custom Action Settings . . . *574*
Setup Files View . . . 576
Prepare for Release View . . . 576
Releases View . . . 576
Express Settings . . . *577*
Media Types . . . *578*
Build Tab . . . *579*
Setup.exe Tab . . . *582*
Signing Tab . . . *589*
.NET/J# Tab . . . *595*
Internet Tab . . . *599*
Events Tab . . . *600*
QuickPatch Projects . . . 602
Define Patch Settings View . . . 602
General Information View . . . *602*
Product Properties . . . 602
Build Settings . . . 603
Common Tab . . . 603
Identification Tab . . . 604
Digital Signature Tab . . . 604
Advanced Tab . . . 607
History . . . 613
Custom Action . . . 613
Files View . . . *613*
New File Settings . . . 613
Modified/Deleted File Settings . . . 614
Registry View . . . *616*
Errors and Warnings . . . 619
Build Errors and Warnings . . . 619
Upgrade Errors and Warnings . . . 663
Windows Installer Run-time Errors . . . 665
Setup.exe Run-Time Errors and Warnings . . . 667
Visual Studio Project Import Errors and Warnings . . . 669
InstallShield Custom Action Reference . . . 689
Command-Line Tools . . . 697
IsCmdBld.exe . . . 697
MsiExec.exe . . . 701

Setup.exe . . . 701
End-User Dialogs. . . . 707
Global Dialog Settings for All End-User Dialogs . . . 707
Splash Bitmap Dialog. . . . 709
Install Welcome Dialog . . . 711
License Agreement Dialog. . . . 712
Readme Dialog. . . . 714
Customer Information Dialog . . . 715
Destination Folder Dialog . . . 718
Database Folder Dialog . . . 719
Setup Type Dialog . . . 721
Custom Setup Dialog . . . 722
Ready to Install Dialog. . . . 724
Setup Progress Dialog . . . 726
Setup Complete Success Dialog . . . 727
MsiRMFilesInUse Dialog . . . 730

10 Frequently Asked Questions . . . 733

Glossary. . . . 735

Index . . . 765

1

InstallShield 2026 Express Edition Help Library

For quick and easy installations that support today's technologies, InstallShield is the industry standard. InstallShield enables you to build reliable installations in record time, and avoid installation-related application errors by keeping current with support for the latest technology.

The InstallShield Help Library contains information about the functionality and features of InstallShield. The Help Library contains the following sections:

Table 1-1 ▪ Help Library Sections

Section	Description
What Was New in Earlier Versions of InstallShield Express Edition	Informs you about changes that were made in earlier versions of InstallShield.
Target System Requirements	Lists the requirements for target systems.
Launching InstallShield with vs. Without Administrative Privileges	Alerts you to functionality that is not available if you are running InstallShield without administrative privileges; also describes a potential problem that may occur if you switch between full Administrator and non-Administrator contexts and you use mapped-drive locations in your projects.
Developing and Building Installations on 32-Bit vs. 64-Bit Systems	Alerts you to differences that you may notice if you use InstallShield on some 32-bit systems compared to some 64-bit systems.
Using Help	Provides information about the InstallShield documentation.
Getting Started	Contains information to help you become familiar with InstallShield, begin creating an installation project, and customize the InstallShield user interface.

Table 1-1 ▪ Help Library Sections (cont.)

Section	Description
Creating Installations	Explains how to create user-friendly, reliable installations and guides you through every step of the process—from specifying information for Add or Remove Programs to building, testing, and deploying an installation.
Updating Applications	Leads you through steps for planning and implementing the various types of upgrades and patches for updating a product. Also explains how you can use FlexNet Connect to notify end users about upgrades and patches that are available.
Additional Installation Options	Discusses a broad range of options available in InstallShield: including the Windows Installer service with an installation, specifying target system requirements for an application, and more.
Integrating InstallShield with External Applications	Contains details about integrating InstallShield with third-party tools such as Microsoft Visual Studio and Microsoft Visual Studio Team Foundation Server (TFS).
Reference	Contains comprehensive reference information for the InstallShield user interface; errors and warnings that might occur when you create, build, or run your installation, and sample end-user dialogs.
Frequently Asked Questions	Directs you to help topics that answer many commonly asked questions about InstallShield and project creation.

Note ▪ *Because the InstallShield Help Library is designed to interact with InstallShield, it is recommended that you open the help from within InstallShield. Copying the help files to another folder or system causes many of its features to work incorrectly.*

For answers to many commonly asked questions and new information about InstallShield that do not appear in the documentation, visit the Knowledge Base.

What Was New in Earlier Versions of InstallShield Express Edition

This section describes features and enhancements that were released in earlier versions of InstallShield Express Edition:

- What's New in InstallShield 2025 Express Edition
- What's New in InstallShield 2023 Express Edition
- What's New in InstallShield 2022 Express Edition

- What's New in InstallShield 2022 Express Edition
- What's New in InstallShield 2021 Express Edition
- What's New in InstallShield 2020 Express Edition
- What's New in InstallShield 2019 Express Edition
- What's New in InstallShield 2018 Express Edition SP1
- What's New in InstallShield 2018 Express Edition
- What's New in InstallShield 2016 SP2 Express Edition
- What's New in InstallShield 2016 SP1 Express Edition
- What's New in InstallShield 2016 Express Edition
- What's New in InstallShield 2015 SP1 Express Edition
- What's New in InstallShield 2015 Express Edition
- What's New in InstallShield 2014 SP1 Express Edition
- What's New in InstallShield 2014 Express Edition
- What's New in InstallShield 2013 SP1 Express Edition
- What's New in InstallShield 2013 Express Edition
- What's New in InstallShield 2012 Spring SP1 Express Edition
- What's New in InstallShield 2012 Spring Express Edition
- What's New in InstallShield 2012 SP1 Express Edition
- What's New in InstallShield 2012 Express Edition
- What's New in InstallShield 2011 Express Edition
- What's New in InstallShield 2010 Express Edition
- What's New in InstallShield 2010 Express Edition SP1
- What's New in InstallShield 2010 Express Edition
- What's New in InstallShield 2009 Express Edition
- What's New in InstallShield 2008 Express Edition
- What's New in InstallShield 12 Express Edition

What's New in InstallShield 2025 Express Edition

InstallShield Express Edition includes the following new feature:

- New Feature in InstallShield 2025 R2 Express Edition

New Feature in InstallShield 2025 R2 Express Edition

For information about new feature added in InstallShield 2025 R2 Express Edition, refer to the following section:

- Enhanced Virtual Machine Support

Enhanced Virtual Machine Support

InstallShield 2025 R2 Express Edition enhances the virtual machine detection capability by introducing support for the following virtual environments where installations can be executed:

- Xen
- Amazon EC2 Instance
- QEMU
- Parallels Desktop
- Google Compute
- Microsoft Azure
- EverRun

This expanded support for virtual machines (mentioned above) enables InstallShield Express Edition to accurately identify the environment in which an installation is running, resulting in more reliable and predictable installation behavior. It allows for thorough compatibility checks across a wide range of virtual platforms, reduces the risk of installation errors, and ensures a seamless, smoother experience for users regardless of the underlying system.

What's New in InstallShield 2023 Express Edition

InstallShield Express Edition includes the following new features:

- New Features in InstallShield 2023 R2 Express Edition
- New Features in InstallShield 2023 R1 Express Edition

New Features in InstallShield 2023 R2 Express Edition

For information about new features added in InstallShield 2023 R2 Express Edition, refer to the following sections:

- Ability to Configure Custom Signing Solution for Digital Signing
- Ability to Store Token Password for EV Digital Signing

Ability to Configure Custom Signing Solution for Digital Signing

InstallShield 2023 R2 Express Edition introduces new signing type settings that enable you to select and configure a custom signing solution to digitally sign build-generated files.

These settings are required only if the customer would like to use custom signing solution instead of the InstallShield Express Edition standard signing solution. Selecting the **Custom** signing type enables additional fields where a custom signing utility path and arguments can be configured. To configure a custom signing solution, you can use the following new settings available in the **Signing** tab for a release in the **Releases** explorer on the **Media** view:

- **Signing Type**—This setting enables you to specify the method to digitally sign build-generated files. The available options for this setting are:
  - **Standard**—Select this option to use the default InstallShield Express Edition sign tool to digitally sign build-generated files.
  - **Custom**—Select this option to use a customized sign tool to digitally sign build-generated files. Selecting this option enables the **Path** and **Arguments** settings.
- **Path**—Specify the sign tool's location to digitally sign build-generated files by using that sign tool. To specify sign tool's location, click the ellipsis button (...) in this setting.
- **Arguments**—Specify the command-line argument for a sign tool's configuration. For example, command-line argument below can be used if the Microsoft built-in signing tool is configured as a custom option to sign the binaries:

  ```
  sign /fd SHA256 /f "<ProgramFilesFolder>\testCA.pfx" /t http://timestamp.digicert.com /p
  MyPassword [filename]
  ```

  Note ▪ *A custom signing tool path uses* `<ProgramFilesFolder>\Windows Kits\10\bin\<WinSDKVer>\x86\signtool.exe`.

  The `[filename]` variable is a place holder for full file path to be signed. It resolves to full path of the binary file to be signed during build time. By default, a file path will be added at the end of an argument and passed to a custom sign tool. Instead of using a hard-coded path, you can use any path variables and environment variables that are defined in your project.

For more information, see the following:

- Signing Tab
- Digital Signature Tab

Note ▪ *By default, the* ***Signing Type*** *setting is set to* ***Standard****.*

Ability to Store Token Password for EV Digital Signing

InstallShield 2023 R2 Express Edition now provides an option to encrypt and store an EV token password in a project file. Prior InstallShield Express Edition versions supported EV certificates via the **Use a certificate store** option. However, when using this option, the user would be asked for the token password each time the file is signed with an EV certificate.

There is also an option called Enable single logon that comes with an EV vendor which would restrict user interventions per session with only one token password request. Configuring the token password in the InstallShield Express Edition signing tab will continue to work untill the password expires. The token password can be changed using the the InstallShield Express Edition IDE.

For more information, see the following:

- Signing Tab
- Digital Signature Tab
- Digitally Sign Setup Dialog Box

New Features in InstallShield 2023 R1 Express Edition

For information about new features added in InstallShield 2023 R1 Express Edition, refer to the following sections:

- New Setup Prerequisites Available
- Ability to Build Setup launcher with Security Features
- Ability to Handle Installers Above 2 Gigabytes

New Setup Prerequisites Available

In the **Redistributables** explorer of the **Application Data** view, InstallShield 2023 R1 Express Edition introduces the following new setup prerequisites:

- Microsoft OLEDB Driver for SQL Server 19.3.0 (x64)
- Microsoft OLEDB Driver for SQL Server 19.3.0 (x86)
- Microsoft .NET 6.0 SDK 6.0.408 (x64)
- Microsoft .NET 6.0 SDK 6.0.408 (x86)
- Microsoft .NET 7.0 SDK 7.0.302 (x64)
- Microsoft .NET 7.0 SDK 7.0.302 (x86)

***Note** ▪ This issue was tracked in ISDEV-43005.*

Ability to Build Setup launcher with Security Features

All InstallShield 2023 R1 Express Edition setup launchers are now built with the the latest security features like control-flow-guard, enforced-code-integrity, and aslr.

***Note** ▪ This issue was tracked in ISDEV-42510.*

Ability to Handle Installers Above 2 Gigabytes

InstallShield 2023 R1 Express Edition enables **LargeAddressAware** flag for all `setup.exe` setup launchers, so that `setup.exe` setup launchers can handle addresses larger than 2 gigabytes during runtime.

***Note** ▪ This issue was tracked in ISDEV-43244.*

What's New in InstallShield 2022 Express Edition

InstallShield Express Edition includes the following new features:

- New Features in InstallShield 2022 R2 Express Edition
- New Features in InstallShield 2022 R1 Express Edition

New Features in InstallShield 2022 R2 Express Edition

For information about new features added in InstallShield 2022 R2 Express Edition, refer to the following section:

Visual Studio 2022 Integration in InstallShield 2022 R2 Express Edition

InstallShield 2022 R2 Express Edition supports creating, editing, upgrading, and building projects in Visual Studio 2022.

The following is a summary of Visual Studio 2022 integration with InstallShield 2022 R2 Express Edition:

- Support for creating new InstallShield Express Edition projects in Visual Studio 2022.
- Support for editing InstallShield Express Edition projects in Visual Studio 2022.
- Support for upgrading projects to the latest version of InstallShield Express Edition.
- Support for building projects from the Visual Studio 2022 IDE and from the x64 MSBuild.

New Features in InstallShield 2022 R1 Express Edition

For information about new features added in InstallShield 2022 R1 Express Edition, refer to the following section:

- Visual Studio 2022 Integration

Visual Studio 2022 Integration

InstallShield 2022 R1 Express Edition now supports integration with Visual Studio 2022. This integration is planned in phases. InstallShield 2022 R1 Express Edition supports to open, upgrade, and build projects in Visual Studio 2022.

Summary of Visual Studio 2022 Integration with InstallShield 2022 R1 Express Edition,

- Support of opening existing projects in Visual Studio 2022
- Upgrade projects to the latest version of InstallShield Express Edition.
- Support of building projects from the Visual Studio 2022 IDE and from the x64 MSBuild.

Note ▪ *Visual Studio 2022 is now being released as a 64-bit application as against 32-bit in previous releases. Hence, the phased delivery of the integration.*

What's New in InstallShield 2021 Express Edition

InstallShield Express Edition includes the following new features:

- New Features in InstallShield Express Edition 2021 R2

- New Features in InstallShield Express Edition 2021 R1

New Features in InstallShield Express Edition 2021 R2

InstallShield Express Edition includes the following enhancement:

- Provision for Installing Files to ARM Specific Folder

Provision for Installing Files to ARM Specific Folder

InstallShield Express edition now supports adding ARM specific folders to the InstallShield project from the Predefined Folder. However, the template summary is not updated to Arm64 in InstallShield Express edition. You should continue with the x86(Intel) Template Summary, which supports copying files to ARM specific folders.

New Features in InstallShield Express Edition 2021 R1

InstallShield Express Edition includes the following new features:

- Support for Windows 11

Support for Windows 11

The Setups created with InstallShield Express Edition 2021 R1 can now run on Windows 11 version.

What's New in InstallShield 2020 Express Edition

InstallShield Express Edition includes the following new features:

- New Features in InstallShield Express Edition 2020 R3
- New Features in InstallShield Express Edition 2020

New Features in InstallShield Express Edition 2020 R3

- InstallShield Azure DevOps Build Extension

InstallShield Azure DevOps Build Extension

InstallShield 2020 R3 introduces extension for building InstallShield projects in Azure DevOps pipelines. To configure the tasks and build InstallShield projects, see InstallShield Azure DevOps Build Extension KB article.

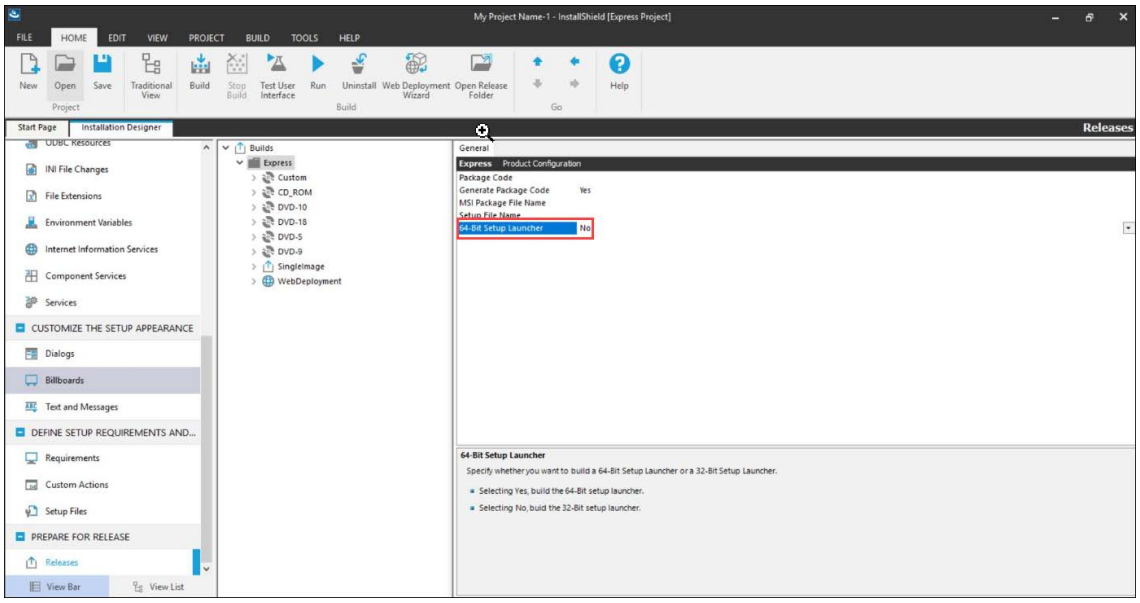
New Features in InstallShield Express Edition 2020

- Pure 64-bit Installers
- Support for AWS CloudHSM Based Digital Signing

Pure 64-bit Installers

Starting from the InstallShield 2020 R1 Express Edition, you will be able to create installers using 64-bit launchers.

To create a Pure 64-Bit Installer, navigate to the Product Configuration view and select 64-Bit Setup Launcher to Yes.



Support for AWS CloudHSM Based Digital Signing

You can now use the InstallShield to digitally sign your installers using an AWS CloudHSM based digital certificate. To enable this feature, add the below property in the file <<InstallShield_Location>/Support/<0409\0411>Settings.xml.

```
<!-- Specify Platform = X86 | X64 for Digital Signing -->
<DigitalSignature Platform="X64"/>
```

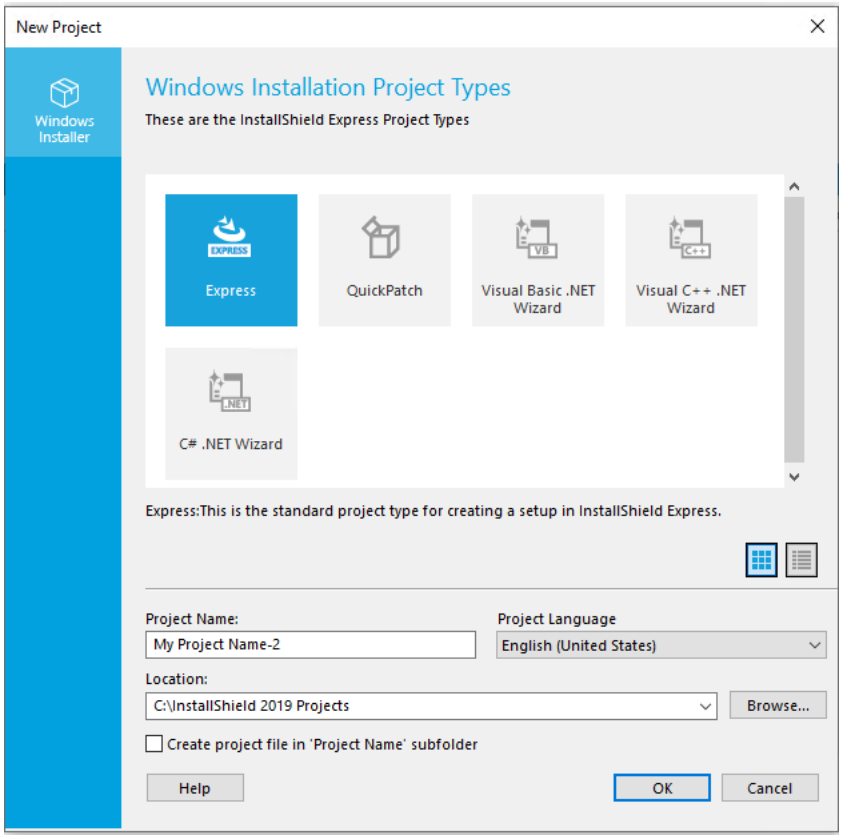
What's New in InstallShield 2019 Express Edition

InstallShield Express Edition includes the following new features:

- New Project Wizard
- Enhancements

New Project Wizard

The New Project Wizard helps you to select different InstallShield Express project types with ease.



You can create different types of projects like Express, QuickPatch, Visual Basic.NET Wizard, Visual C++.NET Wizard, and C#.NET Wizard.

Enhancements

For information about enhancements added in InstallShield 2019 Express Edition, refer to the following section:

- Enhancements in InstallShield 2019 R2 Express Edition
- Enhancements in InstallShield 2019 Express Edition

Enhancements in InstallShield 2019 R2 Express Edition

For information about enhancements added in InstallShield 2019 R2 Express Edition, refer to the following section:

Delay between Signing

InstallShield 2019 R2 Express Edition supports adding a delay between the successive digital signing, this requires only if the timestamp server fails handle the successive signing requests.

Required to specify the <DelayBetweenSigning default="1500"/> node in the settings.xml, under <DevStudio/Build> node in Settings.xml in milliseconds.

Find the Settings.xml file that is installed with InstallShield. Settings.xml is installed in one of the following locations, depending on which language version of InstallShield you are using:

- English—InstallShield Program Files Folder\Support\0409

- Japanese—InstallShield Program Files Folder\Support\0411

Update SQL 2012 Native Client Prerequisites

Project ▪ *This information applies to the following project types:*

- *Basis MSI*
- *InstallScript*
- *InstallScript MSI*

In InstallShield 2019 R2 Express Edition, the Microsoft SQL Server 2012 Native Client prerequisites (x86 and x64) which is now included for the latest versions of 2012 Native Client.

Enhancements in InstallShield 2019 Express Edition

For information about new features added in InstallShield 2019 Express Edition, refer to the following section:

- View Details of a Selected Certificate
- Add Predefined Install Conditions
- Update Default Server to SHA-2 Server
- Add Windows Server 2019 for Operating System Requirement

View Details of a Selected Certificate

In InstallShield 2019 Express Edition, the details of the certificate like the general information of the certificate, security details and certification path is listed on the **View Details** option in the **Certificate Selection** dialog box.

Add Predefined Install Conditions

InstallShield has new predefined system search:

- Microsoft .NET Framework 4.6.2

If your installation requires the above, you can use the System Search view or the Installation Requirements page in the Project Assistant to add this system searche to your project. When end users launch your installation, Windows Installer checks the target system to see if the requirements are met; if they are not met, the installation displays the error message that is defined for the system search.

Update Default Server to SHA-2 Server

In InstallShield 2019 Express Edition, if you select:

- SHA-1 to sign the package, the package will get time timestamped using:
  - <DigitalSignature Timestamp="http://timestamp.verisign.com/scripts/timstamp.dll"/>
- SHA-256 to sign the package, the package will get time timestamped using:
  - <DigitalSignature TimestampRFC3161="http://sha256timestamp.ws.symantec.com/sha256/timestamp"/>

Add Windows Server 2019 for Operating System Requirement

In InstallShield 2019, you can add an option for Windows Server 2019 in the operating system requirements section. You can set the appropriate Install Condition in the project by selecting or deselecting the option.

Additional Prerequisites Included

InstallShield 2019 includes the following additional prerequisites:

Microsoft .Net Core 2.1 Runtime Prerequisite

InstallShield now includes the prerequisites for Microsoft .Net Core 2.1 Runtime in the redistributables view.

What's New in InstallShield 2018 Express Edition SP1

For information about new features added in InstallShield 2018 Express Edition SP1, refer to the following sections:

Support Dual Signing using SHA1 & SHA256 Digest

In previous releases, you could choose the signature digest hashing algorithm only based on:

- Certificate Hash
- SHA-1
- SHA-256

In InstallShield 2018 SP1, along with the above signature digest hashing algorithm, you can now choose Dual Signing - (SHA-1 and SHA-256) digest.

Note ▪ *Using both (SHA1 and SHA256) digests are not supported by the Digital signature of msi.*

What's New in InstallShield 2018 Express Edition

For information about new features added in InstallShield 2018 Express Edition, refer to the following sections:

- Perform Open Source Risk Assessment with FlexNet Code Aware
- Set Forms Authentication on Web Applications

Perform Open Source Risk Assessment with FlexNet Code Aware

InstallShield now includes full integration with FlexNet Code Aware, an automated open source risk assessment and package discovery solution that enables you to quickly scan your products for security and intellectual property (IP) compliance risk.

- Supported File Types
- Running FlexNet Code Aware
- Reading the FlexNet Code Aware Report

Supported File Types

FlexNet Code Aware supports analysis of the following files:

- Java Packages
- Node Packages
- Nuget Packages
- RPM Packages
- Ruby Packages
- EXE & DLL Files

Security vulnerabilities are looked up against the National Vulnerability Database (NVD).

Running FlexNet Code Aware

FlexNet Code Aware is part of InstallShield and no activation ID is required to activate it.

To run FlexNet Code Aware from within InstallShield, click **Scan Project using FlexNet Code Aware** on the InstallShield **Project** menu or click the FlexNet Code Aware icon on the standard toolbar.

Note ▪ *This FlexNet Code Aware menu options are disabled out if you are not currently in an open InstallShield project.*

When FlexNet Code Aware completes the scan of your project, the Results Summary view opens, displaying the number of files scanned, and the number of open-source packages and vulnerabilities found.

When you click the **View Report** button, a full report is displayed.

Reading the FlexNet Code Aware Report

When you click **View Report** on the Results Summary screen, the full FlexNet Code Aware report opens, consisting of an **Initial Summary** view and a **Package Inventory** view.

- **Initial Summary View**—The **Initial Summary** view presents the user with a scan summary, and assessments of operational risk, security vulnerability exposure, and license exposure. The FlexNet Code Aware Initial Summary View displays the following information:
  - **Scan Summary**—This section provides details regarding the codebase that was scanned, including a breakdown of file types, percent of files analyzed, and number of findings.
  - **Operational Risk**—This section provides a composite risk rating based on the combination of packages with Intellectual Property (IP) issues and packages with Security Vulnerabilities.
  - **Security Vulnerability Exposure and License Exposure**—These sections provide a breakdown of the types and categories of identified issues.
- **Package Inventory View**—The **Package Inventory** view, available by clicking **View full package inventory** in the **Scan Summary** section, provides a complete list of discovered open source and third-party packages with associated licenses, security vulnerabilities, dependencies, and detected copyright statements.

  The **Package Inventory** view provides filters that you can use to execute targeted queries to refine the list to various package types of interest.

To view additional package details, click a vulnerability count listed in the **Vulnerabilities** column of the package you want to review. The **Vulnerabilities Detail** page opens (covering a portion of the Package Inventory view), and displays detailed information on the selected package.

Set Forms Authentication on Web Applications

InstallShield 2018 Express Edition includes a new option to set forms authentication on web applications. This new option, **Forms Authentication**, is displayed under the **Authenticated Access** section of the **Internet Information Services** view for a Web site.

Set the **Forms Authentication** option to **Yes** to enable forms authentication. ASP.NET forms-based authentication works well for sites or applications on public Web servers that receive many requests. This authentication mode lets you manage client registration and authentication at the application level, instead of relying on the authentication mechanisms provided by the operating system.

Important ▪ *Forms authentication sends the user name and password to the Web server as plain text. You should use Secure Sockets Layer (SSL) encryption for the Log On page and for all other pages in your application except the Home page.*

Enhancements

InstallShield 2018 Express Edition includes the following enhancements:

- Include the Value of a Property in a Product Configuration's Setup File Name
- New MSBuild Parameters to Set Summary Information Stream Comments and to Set Package File Name
- Additional Prerequisites Included

Include the Value of a Property in a Product Configuration's Setup File Name

In InstallShield 2018 Express Edition, you can now include the value of a property from the Property Table in product release setup and package file names.

For example, you could enter any of the following properties in the **Setup File Name** or **MSI Package File Name** field on the **General** tab of the **Releases > Express** view:

```
setup[ProductVersion]
setup[CustomVersion]
setup[ProductCode]
setup[ProductCode][ProductVersion]
```

If you entered **`setup[ProductVersion]`** in the **Setup File Name** field, it would result in a setup named `setup14.10.1234.exe`, for example.

New MSBuild Parameters to Set Summary Information Stream Comments and to Set Package File Name

In InstallShield 2018 Express Edition, new MSBuild parameters were added to enable you to set add comments to an installer and to set the package file name of an installer.

- New Parameter to Set Summary Information Stream Comments
- New Parameter to Set Package File Name

New Parameter to Set Summary Information Stream Comments

You can add comments to an installer in the **Summary Information Stream Comments** field on the **General Information** view.

In InstallShield 2018 Express Edition, you also have the option of entering comments at build time. A new parameter has been added to the `MSBuild.exe` task, named `SummaryInfoComments`, to set the **Summary Information Stream** comments at build time, such as including the build number, as shown in the following example:

```
MSBuild.exe c:\installers\Setup.sln /Property:SummaryInfoComments="Insert Comments Here"
```

The comments that are added using the `SummaryInfoComments` property can be viewed on the **Properties** dialog box of the built installer.

New Parameter to Set Package File Name

You can specify the package file name of an installer in the **MSI Package File Name** field on the **General** tab on the **Releases > Express** view.

In InstallShield 2018 Express Edition, you also have the option of setting the package file name at build time. A new parameter has been added to the `MSBuild.exe` task, named `MSIPackageFileName`, to set the package file name of the built installer at build time, as shown in the following example:

```
MSBuild.exe c:\installers\Setup.isproj /Property:MSIPackageFileName="MySetup"
```

When entering the value for the `MSIPackageFileName` parameter, you need to enter the file name—without the period or the file extension—that InstallShield should use for the `.msi` file.

Additional Prerequisites Included

InstallShield 2018 Express Edition includes the following additional prerequisites:

- Visual C++ 2017 x86 and x64 Prerequisites
- Microsoft SQL Server 2014 SP1 and SP2 Prerequisites
- Microsoft .NET Framework 4.7 Prerequisite

Visual C++ 2017 x86 and x64 Prerequisites

Because Microsoft Visual Studio 2017 has been released, InstallShield now includes the prerequisites for Visual C++ 2017 x86 and x64.

Microsoft SQL Server 2014 SP1 and SP2 Prerequisites

Because Microsoft SQL Server 2014 has had 2 Service Packs released, InstallShield now includes the prerequisites for both Microsoft SQL Server 2014 SP1 and SP2.

Microsoft .NET Framework 4.7 Prerequisite

In InstallShield 2018 now includes a prerequisite for Microsoft .NET Framework 4.7.

In InstallShield 2018 R2, you can now add pre-defined install conditions for .net 4.7:

- 4.7
- 4.7.1
- 4.7.2

What's New in InstallShield 2016 SP2 Express Edition

New Features

Integration with FlexNet Code Aware

InstallShield now includes integration with FlexNet Code Aware, an automated open source risk assessment and package discovery solution that enables you to quickly scan your products for security and intellectual property (IP) compliance risk.

The current release of FlexNet Code Aware supports analysis of the following files:

- Java Packages
- Node Packages
- Nuget Packages
- RPM Packages
- Ruby Packages
- EXE & DLL Files

Security vulnerabilities are looked up against the National Vulnerability Database (NVD).

Running FlexNet Code Aware

FlexNet Code requires a separate license from InstallShield. There is also trial/evaluation version. For more information, refer to the FlexNet Code Aware product page of the Revenera Web site.

To run FlexNet Code Aware from within InstallShield, click **Scan Project using FlexNet Code Aware** from the InstallShield **Project** menu. This menu option is disabled out if you are not currently in an open InstallShield project. A FlexNet Code Aware icon is also available on the InstallShield standard toolbar.

When FlexNet Code Aware completes the scan of your project, a summary displays showing the number of files scanned, and the number of open-source packages and vulnerabilities found. A **View report** button is provided if you have a fully licensed version of FlexNet Code Aware. For more information about the details provided in this report, refer to Reading the FlexNet Code Aware Report.

Reading the FlexNet Code Aware Report

Note ▪ *The FlexNet Code Aware Report is not available in trial/evaluation mode. A fully licensed version of FlexNet Code Aware is required.*

To view the FlexNet Code Aware Report, click **View report** on the summary dialog that appears after FlexNet Code Aware has scanned your project.

The FlexNet Code Aware report consists of several sections:

- The initial Summary View presents the user with a **Scan Summary**, **Operational Risk** assessment, **Security Vulnerability Exposure,** and **License Exposure**.
  - The **Scan Summary** section provides details regarding the codebase that was scanned, including a breakdown of file types, percent of files analyzed, and number of findings.

  - The **Operational Risk** section provides a composite risk rating based on the combination of packages with Intellectual Property (IP) issues and packages with Security Vulnerabilities.
  - The **Security Vulnerability Exposure** and **License Exposure** sections provide a breakdown of the types and categories of identified issues.
- The Package Inventory View, available by clicking **view full package inventory** in the **Scan Summary** section, provides a complete list of discovered open source and third-party packages with associated licenses, security vulnerabilities, dependencies, and detected copyright statements.

  The Package Inventory View provides filters that you can use to execute targeted queries to refine the list to various package types of interest.

Viewing Package Details

Click a vulnerability count listed in the Vulnerabilities column of the Package Inventory report page for each package you want to review. The **Vulnerabilities detail** page appears, covering a portion of the Package Inventory report.

What's New in InstallShield 2016 SP1 Express Edition

New Features

Support for Microsoft Visual Studio 2017

InstallShield 2016 Express Edition includes support for Visual Studio 2017. You can create InstallShield 2016 Express Edition projects from within this version of Visual Studio.

What's New in InstallShield 2016 Express Edition

New Features

InstallShield 2026 Express Edition includes the following new features:

- Support for the Latest Releases of Windows Operating Systems
- Tile Configurations
- New InstallShield Prerequisites for Microsoft Visual C++ 2015, .NET Framework 4.6, and More
- Predefined System Searches for Adobe Reader, Microsoft Office and the .NET Framework

Support for the Latest Releases of Windows Operating Systems

InstallShield 2016 supports the latest releases of the Windows operating system:

- Windows 10 Anniversary Update
- Windows Server 2016

Not only can you install InstallShield on these operating systems, but you can also create installers that target these operating systems.

Tile Configurations

Windows 8 introduced a grid of application tiles to the Start screen, replacing the usual list of shortcuts, and also presented tiles in place of shortcuts. InstallShield supports customizing the appearance of a desktop app's tile on the Start screen. The following tile configuration settings are available:

- A toggle between light or dark text when including the app name on medium-sized (150x150) tiles
- Choice of tile background color
- Option to use custom tile images (small: 70x70 and medium:150x150)
- Preference to show or hide the app name on medium-sized tiles

The **Tile Configurations** node appears in the main **Shortcuts/Folder** view. Any applicable tile configurations are listed.

To learn more, see the following topics:

- Configuring the Appearance of a Desktop App's Tile on the Start Screen
- Tile Configuration Settings

New InstallShield Prerequisites for Microsoft Visual C++ 2015, .NET Framework 4.6, and More

InstallShield includes the following InstallShield prerequisites that you can add to projects:

- Microsoft .NET Framework 4.6.2 Full
- Microsoft .NET Framework 4.6.1 Full
- Microsoft .NET Framework 4.6.1 Web
- Microsoft ReportViewer 2015
- Microsoft SQL Server 2014 Express System CLR Types (x86)
- Microsoft SQL Server 2016 Express RTM (x64)
- Microsoft SQL Server 2016 Express RTM LocalDB (x64)
- Microsoft Visual C++ 2015 Update 3 Redistributable Package (x86)
- Microsoft Visual C++ 2015 Update 3 Redistributable Package (x64)
- Windows Management Framework 4.0 for Windows 7 SP1 and Server 2008 R2 SP1 (x64)
- Windows Management Framework 4.0 for Windows Server 2012 (x64)
- Windows Management Framework 5.0 for Windows 7 SP1 (x86)
- Windows Management Framework 5.0 for Windows 7 SP1 and Server 2008 R2 SP1 (x64)
- Windows Management Framework 5.0 for Windows 8.1 (x86)
- Windows Management Framework 5.0 for Windows 8.1 and Server 2012 R2 (x64)
- Windows Management Framework 5.0 for Windows Server 2012 (x64)

These prerequisites install the appropriate technologies on supported target systems.

Note ▪ *The* **Web** *prerequisite for the .NET Framework requires an Internet connection. This prerequisite downloads the required redistributable files if appropriate. The* **Full** *prerequisite for the .NET Framework is a stand-alone installation that does not require an Internet connection.*

Predefined System Searches for Adobe Reader, Microsoft Office and the .NET Framework

InstallShield has new predefined system searches:

- Adobe Reader 11
- Adobe Reader DC
- Microsoft Office 2013
- Microsoft Office 2016
- Microsoft .NET Framework 4.5.1
- Microsoft .NET Framework 4.5.2
- Microsoft .NET Framework 4.6
- Microsoft .NET Framework 4.6.1
- Microsoft .NET Framework 4.6.2

If your installation requires one or more of these, you can use the Requirements view or the Installation Requirements page in the Project Assistant to add these system searches to your project. When end users launch your installation, Windows Installer checks the target system to see if the requirements are met; if they are not met, the installation displays the error message that is defined for the system search.

Enhancements

InstallShield 2016 includes the following new enhancements:

- Ability to Filter Items by Features in Additional Views
- Digital Signing Improvements

Ability to Filter Items by Features in Additional Views

The following views now contain a View Filter that lets you filter the view list by any feature in your project:

- **Environment Variables View**—You can use the View Filter list at the top of this view to show and hide environmental variables that are associated to a particular feature in your project. You can also select a feature from the View List in order to associate only that feature with a subsequent event (e.g., the creation, modification, or removal of an environmental variable). Lastly, to see all of the environmental variables that are in your project, select the All Application Data option in the View Filter list. For more information, see Environment Variables View.
- **INI File Changes View**—You can use the View Filter list at the top of this view to show and hide initialization (.ini) files that are associated to a particular feature in your project. You can select a feature from the View List in order to associate only that feature with a subsequent event (e.g., the creation, importing, modification, or removal of .ini files). The resulting modification takes place at run time on the target system when the feature is installed. Lastly, to see all of the .ini files that are in your project, select the All Application Data option in the View Filter list. For more information, see INI File Changes View.

Digital Signature Updates

Beginning with InstallShield 2015, support was added to enable you to use digital certificates that use the SHA-256 hashing algorithm for signing your installations and files at build time.

In InstallShield 2016, support for SHA-256 digital certificates has been enhanced for Windows Installer and InstallScript projects to:

- Give you the ability to specify a digest type using the new **Signature Digest** drop-down on the Certificate Selection Dialog Box
- RFC3161 timestamping is now supported and can be specified in `settings.xml`, noting that:
  - **DigitalSignature/@Timestamp** can be an **Authenticode** or **RFC3161 server** for .msi, .exe, and .dll files
  - **DigitalSignature/@TimestampRFC3161** used for UWP app package files must be an **RFC3161 server**
- Handle similarly-named certificates in the Certificate Store

In InstallShield 2019 Express Edition, the timestamp server is set to a SHA-2 server to:

- <DigitalSignature Timestamp="http://sha256timestamp.ws.symantec.com/sha256/timestamp"/>

Important ▪ *Any new signatures created or timestamped after Jan 1, 2016 must be SHA-256-based signatures. Any files signed with an SHA-1 certificate need to have a timestamp showing a date and time prior to Jan 1, 2016 in order to continue to be supported. Those files will still be allowed through the 'Mark-of-the-web" system until Jan 14, 2020, when all SHA-1 support will stop in all current versions of Windows.*

What's New in InstallShield 2015 SP1 Express Edition

New Features

InstallShield includes the following new features.

Support for Windows 10

InstallShield has support for Windows 10.

Support for Microsoft Visual Studio 2015

InstallShield includes support for Visual Studio 2015. You can create InstallShield projects from within this version of Visual Studio.

New InstallShield Prerequisites for Microsoft Visual C++ 2015 and .NET Framework 4.6

InstallShield includes new InstallShield prerequisites that you can add to your projects:

- Microsoft Visual C++ 2015 Redistributable Package (x86)
- Microsoft Visual C++ 2015 Redistributable Package (x64)
- Microsoft .NET Framework 4.6 Full
- Microsoft .NET Framework 4.6 Web

These prerequisites install the appropriate technologies on supported target systems.

What's New in InstallShield 2015 Express Edition

New Features

InstallShield includes the following new features.

Support for Windows 10–Based Systems

InstallShield has support for Windows 10.

Targeting Windows 10

On systems with Windows 10, the Windows Installer properties `VersionNT` and `VersionNT64` indicate 603, which was originally introduced as the version number of Windows 8.1. Therefore, it is not possible to create conditions in an .msi package that specifically target Windows 10.

Since Windows Installer 5.0 and Windows 7, DLL custom actions in .msi packages are shimmed to block obtaining the operating system version; the APIs **GetVersion**, **GetVersionEx**, and **RtlGetVersion** return a Windows version of 6.0.6000, which was originally the version number of Windows Vista. Therefore, it is also not possible to obtain the actual version number of Windows from a DLL custom action.

Because of the aforementioned behavior in Windows Installer, it is not easily possible to detect what version of Windows on which an .msi package is running. In areas where you can specify target system OS requirements, such as the Installation Requirements page in the Project Assistant, or the Requirements view, the Windows 8.1 option has been renamed as **Windows 8.1 or Windows 10** to reflect the new run-time behavior.

The InstallShield prerequisites that should be installable on Windows 10 have been updated so that they are installed on those systems if needed. Previously, the prerequisites may not have run by default on those systems.

Support for Microsoft Visual Studio 2015

InstallShield includes support for Visual Studio 2015. You can create InstallShield projects from within this version of Visual Studio.

Digital Signing Improvements

InstallShield includes several improvements for digitally signing your installations and files at build time.

Support for SHA-256 Digital Certificates

InstallShield now enables you to use digital certificates that use the SHA-256 hashing algorithm for signing your installations and files at build time.

SHA-256 is favored over SHA-1, which is being deprecated because of the potential for security vulnerabilities. Microsoft announced that Windows will stop trusting items that were signed and timestamped with SHA-1 certificates after January 1, 2016. In addition, certification authorities—the organizations that issue certificates—are phasing out the creation of SHA-1 certificates. Thus, it is recommended that you replace any SHA-1 certificates in your InstallShield projects with SHA-256 certificates. For the latest information and more specific details, check with your certification authority.

In InstallShield, to replace a SHA-1 certificate with a SHA-256 certificate for signing your releases, use the Signing tab in the Releases view to replace the reference to the current certificate with one for a SHA-256 certificate.

If your project is configured to sign with a SHA-256 certificate, InstallShield uses a SHA-256 hash in the signature of the files that it signs at build time. If your project is still configured to sign with a SHA-1 certificate, InstallShield uses a SHA-1 hash; in addition, use of a SHA-1 certificate now triggers build warning -7346 to alert you about the SHA-1 usage.

In earlier versions of InstallShield, InstallShield used a SHA-1 hash in the signature of files when signing with either SHA-1 and SHA-256 certificates.

For details, see Digital Signing and Security.

Ability to Use a Certificate Store for Referencing Certificates

When you are specifying the digital signature information that you want to use for signing your files and installations, InstallShield now lets you reference a certificate store that contains the certificate that you want to use. This support is available as an alternative to specifying a .pfx certificate file on your machine.

To specify whether you want to use a certificate store or a .pfx certificate, use the Digital Certificate File setting on the Signing tab in the Releases view. When you click the ellipsis button (...) in this setting, a new Certificate Selection dialog box opens, enabling you to specify certificate information such as the store name (Personal, Trusted Root Certification Authorities, Enterprise Trust, Intermediate Certification Authorities), the store location (user or machine), and the subject that identifies the specific certificate that you want to use. As an alternative, you can specify on this dialog box the path and file name of a .pfx file that you want to use.

If you configure your project to use a certificate that was imported with password protection into a store, Windows prompts for the password at build time when InstallShield is attempting to sign your project's files. The strong key protection that Windows uses does not permit InstallShield to provide the password to the cryptographic provider.

Certificate store support is also available for signing QuickPatch packages. To specify certificate store or .pfx certificate information for a QuickPatch package, use the Build Settings area of the General Information view in the QuickPatch project. This area includes a Digital Signature tab that includes the new support.

To learn more, see:

- Digital Signing and Security
- Certificate Selection Dialog Box
- Signing Tab (for a release in the Releases view)
- Digital Signature Tab (in a QuickPatch project)

Note that InstallShield no longer has support for signing with .spc and .pvk files. To learn how to convert those files to a .pfx file, see Digital Signing and Security.

Ability to Specify a Program Name for the UAC Dialog Box

The Signing tab in the Releases view includes a new Signature Description setting. Use this setting to specify the text that you want to be displayed to the right of the "Program Name:" label on the UAC dialog box for the Setup.exe files, the .msi file, and other installation files that InstallShield signs at build time. The UAC dialog box opens when an end user launches the signed file and elevated privileges are required.

If you leave the Signature Description setting blank, InstallShield uses the name of the file without its extension as the text on the UAC dialog box.

For more information, see Signing Tab.

Ability to View Both the 32- and 64-Bit Areas of the Source Machine's Registry on 64-Bit Development Systems

If you are using InstallShield on a 64-bit development system, the Registry view in InstallShield now displays both the 32-bit and 64-bit areas of your machine's registry:

- HKEY_LOCAL_MACHINE\Software
- HKEY_LOCAL_MACHINE\Software\Wow6432Node

This support makes it easier to develop installations on 64-bit machines, since it enables you to drag and drop entries from those source areas to the appropriate areas in the destination pane of this view.

Previously, if you were using InstallShield on a 64-bit development system, the source panes in the Registry view in InstallShield did not show any 64-bit data from the HKLM\Software part of the registry; in addition, the source panes displayed 32-data from the machine's HKLM\Software\Wow6432Node area in the HKLM\Software area.

Note that if you want your installation to install registry data to a 64-bit area of the registry on 64-bit target systems without having it redirected to a 32-bit area, you must place the registry data in the HKEY_LOCAL_MACHINE\SOFTWARE (64-Bit) node in the destination pane in the Registry view. Simply dragging 64-bit data from the source panes in the Registry view to a non-64-bit location in one of the destination panes of the view does not mark the component as 64 bit.

For more information, see:

- Developing and Building Installations on 32-Bit vs. 64-Bit Systems
- Dragging and Dropping Registry Entries to Create Registry Keys
- Registry View

New InstallShield Prerequisites for Microsoft Visual C++ 2015, .NET Framework 4.6, and More

InstallShield includes new InstallShield prerequisites that you can add to your projects:

- Microsoft Visual C++ 2015 Redistributable Package (x64)
- Microsoft Visual C++ 2015 Redistributable Package (x86)
- Microsoft Visual C++ 2013 Redistributable Package (x86)
- Microsoft Visual C++ 2013 Redistributable Package (x64)
- Microsoft .NET Framework 4.6 Full
- Microsoft .NET Framework 4.6 Web
- Microsoft .NET Framework 4.5.2 Full
- Microsoft .NET Framework 4.5.2 Web
- Microsoft SQL Server 2012 Express SP2 (x86)
- Microsoft SQL Server 2012 Express SP2 (x86 & x64Wow)
- Microsoft SQL Server 2012 Express SP2 (x64)

- Microsoft SQL Server 2012 Express SP2 LocalDB (x86)
- Microsoft SQL Server 2012 Express SP2 LocalDB (x64)
- Microsoft SQL Server 2012 Express SP2 Management Objects (x86)
- Microsoft SQL Server 2012 Express SP2 Management Objects (x64)
- Microsoft SQL Server 2012 Express SP2 System CLR Types (x86)
- Microsoft SQL Server 2012 Express SP2 System CLR Types (x64)
- Internet Explorer 11.0 for Windows 7 (x86)
- Internet Explorer 11.0 for Windows 7 and Windows Server 2008 R2 (x64)
- Microsoft ReportViewer 2012

These prerequisites install the appropriate technologies on supported target systems.

The Microsoft SQL Server 2012 Express SP2 prerequisites replace the Microsoft SQL Server 2012 Express SP1 prerequisites.

New Predefined System Searches for Internet Explorer 10 and 11

InstallShield has new predefined system searches that check target systems for Internet Explorer 10 or Internet Explorer 11. If your installation or product requires either of those versions, you can use the Requirements view or the Installation Requirements page in the Project Assistant to add one of these system searches to your project. When end users launch your installation, Windows Installer checks the target system to see if the requirements are met; if they are not met, the installation displays the error message that is defined for the system search.

Enhancements

Performance Improvement for the Files and Folders View

InstallShield has been enhanced to more quickly load the Files view of large projects.

What's New in InstallShield 2014 SP1 Express Edition

InstallShield includes the following improvements.

Updates to Portuguese Run-Time Strings

The default run-time strings for Portuguese (Brazil) and Portuguese (Portugal) have been updated to reflect the Portuguese Language Orthographic Agreement of 1990 that finishes going into effect at the end of 2014, per a 6-year transition period starting in 2009.

What's New in InstallShield 2014 Express Edition

New Features

InstallShield includes the following new features.

New InstallShield Prerequisites for .NET Framework 4.5.1

InstallShield includes new InstallShield prerequisites that you can add to projects:

- Microsoft .NET Framework 4.5.1 Full
- Microsoft .NET Framework 4.5.1 Web

These prerequisites install the .NET Framework 4.5.1 on supported target systems.

Microsoft SQL Server 2014 Prerequisites

InstallShield includes several new SQL Server 2014–related InstallShield prerequisites that you can add to projects:

- Microsoft SQL Server 2014 Express RTM (x64)
- Microsoft SQL Server 2014 Express RTM (x86 & x64Wow)
- Microsoft SQL Server 2014 Express RTM (x86)
- Microsoft SQL Server 2014 Express RTM LocalDB (x64)
- Microsoft SQL Server 2014 Express RTM LocalDB (x86)

These InstallShield prerequisites install the technology on supported target systems.

New InstallShield Prerequisites for Microsoft Visual C++ 2012 Update 4

InstallShield includes new InstallShield prerequisites that you can add to projects:

- Microsoft Visual C++ 2012 Update 4 Redistributable Package (x86)
- Microsoft Visual C++ 2012 Update 4 Redistributable Package (x64)

These prerequisites install the various technologies on supported target systems.

Ability to Include Support Files in Subfolders

InstallShield now lets you specify custom folder structures for support files.To add a subfolder under one of the nodes in the Setup Files view, right-click the node and then click New Folder. InstallShield adds a subfolder and enables you to rename it as needed. You can also add a nested folder structure. To add files to a subfolder, select it and right-click in the right pane, and then click Insert Files. At run-time, the installation copies the support folders and files to `SUPPORTDIR`, a temporary directory on target systems, to make them available during your product's installation process. The support folders and files are deleted when the installation is complete.

To learn more, see Adding Setup Files.

Support for Removing Files and Folders

InstallShield now has built-in support that makes it easy to specify files and folders that you want to be removed from target systems at run time. This file and folder removal capability is useful for scenarios such as removing application-created files that your installation does not otherwise track.

You can schedule the removal of a file or folder for one of the following events:

- When the file or folder's feature is being installed
- When the file or folder's feature is being uninstalled

- When the file or folder's feature is being installed or uninstalled

Note that if the item to be removed is a folder, that folder is removed only if it is empty.

To configure a file or folder removal in a project, use the Files view. In this view, select the destination folder that contains the file or folder that you want to be removed. Then right-click in the **Destination computer's files** pane and click Add File Removal. InstallShield displays a Properties dialog box that lets you configure the available removal settings.

To learn more, see:

- Removing Files and Folders from Target Systems
- File Removal Properties Dialog Box

Enhancements

InstallShield includes the following enhancements.

Support for Offering Printer Selection for the License Agreement Dialog at Run Time

The behavior of the Print button on License Agreement dialogs has been enhanced. Instead of printing directly to the default printer when an end user clicks the Print button, the printer selection dialog box now opens.

64-Bit Support for Registry-Related System Searches

The System Search Wizard enables you to define searches that you want Windows Installer to perform; Windows Installer can search for a particular file, folder, registry key or .ini value on target systems. The registry-related panel in the System Search Wizard includes a new check box that lets you specify whether you want to check the 64-bit area of the registry on 64-bit target systems.

- System Search Wizard
- How do you want to look for it? Panel (Registry Search Options)
- Challenges of Supporting Both 32-Bit and 64-Bit Target Operating Systems

New FlexNet Connect 13.06 Redistributables Available

InstallShield includes support for FlexNet Connect 13.06 in your projects. Use the Update Notifications view in InstallShield to include one of the two FlexNet Connect 13.06 merge modules—one has the Common Software Manager, and the other does not.

What's New in InstallShield 2013 SP1 Express Edition

InstallShield 2013 Service Pack 1 (SP1) includes changes that offer support for the final released versions of Windows 8.1, Windows Server 2012 R2, and Visual Studio 2013.

Ability to Target Windows 8.1 and Windows Server 2012 R2 Systems

InstallShield enables you to specify that your installation requires Windows 8.1 or Windows Server 2012 R2. It also lets you build feature conditions for these operating systems.

The InstallShield prerequisites that should be installable on Windows 8.1 and Windows Server 2012 R2 have been updated so that they are installed on those systems if needed. Previously, the prerequisites were not run by default on those systems. This applies to the following InstallShield prerequisites:

- FSharp Redistributable Package 2.0
- JRE_SE 1.7.0_02 (x64)
- JRE_SE 1.7.0_02 (x86)
- Microsoft .NET Framework 3.0 OS Component
- Microsoft .NET Framework 3.5 SP1 (Windows Feature)
- Microsoft .NET Framework 4.5 Full
- Microsoft .NET Framework 4.5 Web
- Microsoft App-V 5.0 SP1 Desktop Client (x64)
- Microsoft App-V 5.0 SP1 Desktop Client (x86)
- Microsoft ReportViewer 2010
- Microsoft SQL CE 3.5 SP2
- Microsoft SQL Server 2005 Express SP3 (x86 & x64Wow)
- Microsoft SQL Server 2005 Express SP3 (x86)
- Microsoft SQL Server 2008 Express SP1 (x64)
- Microsoft SQL Server 2008 Express SP1 (x86 & x64Wow)
- Microsoft SQL Server 2008 Express SP1 (x86)
- Microsoft SQL Server 2008 Management Objects 10.00.2531 (IA64)
- Microsoft SQL Server 2008 Management Objects 10.00.2531 (x64)
- Microsoft SQL Server 2008 Management Objects 10.00.2531 (x86)
- Microsoft SQL Server 2008 Native Client 10.00.2531 (IA64)
- Microsoft SQL Server 2008 Native Client 10.00.2531 (x64)
- Microsoft SQL Server 2008 Native Client 10.00.2531 (x86)
- Microsoft SQL Server 2008 R2 Express RTM (x64)
- Microsoft SQL Server 2008 R2 Express RTM (x86 & x64Wow)
- Microsoft SQL Server 2008 R2 Express RTM (x86)
- Microsoft SQL Server 2008 R2 Express SP2 (x64)
- Microsoft SQL Server 2008 R2 Express SP2 (x86 & x64Wow)
- Microsoft SQL Server 2008 R2 Express SP2 (x86)
- Microsoft SQL Server 2008 R2 Native Client 10.50.1600.1 (IA64)
- Microsoft SQL Server 2008 R2 Native Client 10.50.1600.1 (x64)

- Microsoft SQL Server 2008 R2 Native Client 10.50.1600.1 (x86)
- Microsoft SQL Server 2012 Express LocalDB RTM (x64)
- Microsoft SQL Server 2012 Express LocalDB RTM (x86)
- Microsoft SQL Server 2012 Express RTM (x64)
- Microsoft SQL Server 2012 Express RTM (x86 & x64Wow)
- Microsoft SQL Server 2012 Express RTM (x86)
- Microsoft SQL Server 2012 Native Client (x64)
- Microsoft SQL Server 2012 Native Client (x86)
- Microsoft SQL Server Compact 4.0 (x64)
- Microsoft SQL Server Compact 4.0 (x86)
- Microsoft SQL Server Native Client 9.00.4035 (IA64)
- Microsoft SQL Server Native Client 9.00.4035 (x64)
- Microsoft SQL Server Native Client 9.00.4035 (x86)
- Microsoft SQL Server System CLR Types 10.00.2531 (IA64)
- Microsoft SQL Server System CLR Types 10.00.2531 (x64)
- Microsoft SQL Server System CLR Types 10.00.2531 (x86)
- Microsoft Visual C++ 2005 SP1 Redistributable MFC Security Update KB2538242(x64)
- Microsoft Visual C++ 2005 SP1 Redistributable MFC Security Update KB2538242(x86)
- Microsoft Visual C++ 2005 SP1 Redistributable Package (x64)
- Microsoft Visual C++ 2005 SP1 Redistributable Package (x86)
- Microsoft Visual C++ 2008 SP1 Redistributable MFC Security Update KB2538243(x64)
- Microsoft Visual C++ 2008 SP1 Redistributable MFC Security Update KB2538243(x86)
- Microsoft Visual C++ 2008 SP1 Redistributable Package (x64)
- Microsoft Visual C++ 2008 SP1 Redistributable Package (x86)
- Microsoft Visual C++ 2010 Redistributable Package (x64)
- Microsoft Visual C++ 2010 Redistributable Package (x86)
- Microsoft Visual C++ 2010 RTM Redistributable MFC Security Update KB2467173 (x64)
- Microsoft Visual C++ 2010 RTM Redistributable MFC Security Update KB2467173 (x86)
- Microsoft Visual C++ 2010 SP1 Redistributable Package (x64)
- Microsoft Visual C++ 2010 SP1 Redistributable Package (x86)
- Microsoft Visual C++ 2012 Redistributable Package (x64)
- Microsoft Visual C++ 2012 Redistributable Package (x86)
- Microsoft Visual C++ 2012 Update 1 Redistributable Package (x64)

- Microsoft Visual C++ 2012 Update 1 Redistributable Package (x86)
- Microsoft VSTO 2010 Runtime (x64)
- Microsoft VSTO 2010 Runtime

Support for Microsoft Visual Studio 2013

InstallShield includes support for Visual Studio 2013. You can create InstallShield projects from within this version of Visual Studio.

New InstallShield Prerequisites for Microsoft SQL Server 2012 Express SP1

InstallShield includes several new SQL Server–related InstallShield prerequisites that you can add to projects:

- Microsoft SQL Server 2012 Express SP1 LocalDB (x64)
- Microsoft SQL Server 2012 Express SP1 LocalDB (x86)
- Microsoft SQL Server 2012 Express SP1 (x64)
- Microsoft SQL Server 2012 Express SP1 (x86 & x64Wow)
- Microsoft SQL Server 2012 Express SP1 (x86)
- Microsoft SQL Server 2012 Express SP1 Management Objects (x64)
- Microsoft SQL Server 2012 Express SP1 Management Objects (x86)
- Microsoft SQL Server 2012 Express SP1 System CLR Types (x64)
- Microsoft SQL Server 2012 Express SP1 System CLR Types (x86)

These InstallShield prerequisites install the technology on supported target systems.

New Predefined Path Variable for the Visual Studio Solution Folder

A new predefined path variable called VSSolutionFolder is available in projects to reference a higher-level base directory. This support enables you to have in your InstallShield projects static links to files in sibling projects that are within the Visual Studio solution folder; if you work on the projects on a different machine, the static links that use the VSSolutionFolder path variable can reference the correct paths for the files in sibling projects.

The VSSolutionFolder path variable is defined automatically whenever an InstallShield project is opened from within a Visual Studio solution. It is also defined automatically if you are using MSBuild to build a solution that contains an InstallShield project. However, in other scenarios, when the InstallShield project is opened without the Visual Studio solution, VSSolutionFolder cannot be defined automatically. For example, if you open the InstallShield project in InstallShield directly, without having Visual Studio open, VSSolutionFolder is not defined. Similarly, if you use the command-line tool `IsCmdBld.exe`, or if you use MSBuild with an .isproj file, VSSolutionFolder is not defined. If you are using MSBuild to build a release in InstallShield project, use the PathVariables parameter to set the value of VSSolutionFolder. This parameter is exposed as the ItemGroup InstallShieldPathVariableOverrides when the default targets file is used.

If you include in your InstallShield project a source file whose path includes the VSSolutionFolder path variable and build it in an environment that does not support the VSSolutionFolder path variable, build errors such as the following ones may occur:

- -6103: Could not find file <VSSolutionFolder>\MyFile.exe

- -6271: File <VSSolutionFolder>\MyFile.exe not found. An error occurred building the MsiFileHash table record for this file. Verify that the file exists in the specified location.

To learn more, see:

- Using the VSSolutionFolder Path Variable with Visual Studio Solutions
- Adding Files and Folders to a Project
- Using MSBuild to Build a Release from the Command Line

What's New in InstallShield 2013 Express Edition

New Features

InstallShield includes the following new features.

New InstallShield Prerequisites for .NET Framework 3.5 SP1, Microsoft Visual C++ 2012, and SQL Server 2008 R2 Express SP2

InstallShield includes the following InstallShield prerequisites that you can add to projects:

- Microsoft .NET Framework 3.5 SP1 (Windows Feature)
- Microsoft SQL Server 2008 R2 Express SP2 (x64)
- Microsoft SQL Server 2008 R2 Express SP2 (x86 & x64Wow)
- Microsoft SQL Server 2008 R2 Express SP2 (x86)
- Microsoft Visual C++ 2012 Update 1 Redistributable Package (x64)
- Microsoft Visual C++ 2012 Update 1 Redistributable Package (x86)

These prerequisites install the various technologies on supported target systems.

Ability to Install and Start Windows Services

InstallShield now includes support for installing a service during installation, and removing the service during uninstallation. It also now has support for optionally starting the service after installing it, starting it automatically every time that the system starts, or starting it on demand (when the service is requested through the Service Control Manager).

To configure information about a service in your project, use the new Services view.

To learn more, see the following:

- Installing and Configuring Windows Services
- Services View

Ability to Create Installations that Install to 64-Bit Locations

Microsoft designed 64-bit versions of Windows to allow existing 32-bit applications to continue to work seamlessly. They also designed 64-bit versions of Windows in such a way to allow a recompiled version of the same code to work seamlessly as a 64-bit application. To provide this support, 64-bit versions of Windows isolate the 32-bit and 64-bit portions from each other in two main ways: their files are stored in separate locations (for

example: Program Files vs. Program Files (x86); System32 vs. SysWow64), and their registry keys are separated (HKLM\Software vs. HKLM\Software\Wow6432Node). Thus, if end users try to run a 32-bit Windows Installer–based installation on a 64-bit system, the files, folders, and registry entries that are configured to be installed to locations such as Program Files, System32, and HKLM\Software are redirected to Program Files (x86), SysWow64, HKLM\Software\Wow6432Node, respectively.

The Express edition of InstallShield now has support for creating 64-bit packages that can target 64-bit systems and install to 64-bit locations—for example, Program Files instead of Program Files (x86), System32 instead of SysWow64, and HKLM\Software instead of HKLM\Software\Wow6432Node. Previously, this support was available only in the Premier and Professional editions of InstallShield.

To enable this 64-bit support, the following changes have been made in the Express edition of InstallShield:

- The Files view now contains predefined folders for 64-bit locations. To specify that a file or folder should be installed to a 64-bit folder, add the file or folder to the appropriate predefined 64-bit folder. For example, to install a folder to the 64-bit Program Files Folder on 64-bit systems, add the folder to the new [ProgramFiles64Folder] node in this view. Note that 64-bit folders are not displayed by default. To display a 64-bit folder in this view: Right-click a folder in the **Destination computer's folders** pane, point to Show Predefined Folders, and then click [ProgramFiles64Folder].
- The Registry view now has support for 64-bit registry locations. The SOFTWARE registry entry in the **Destination computer's Registry view** pane in this view has been split into two separate nodes: SOFTWARE (32-Bit) and SOFTWARE (64-Bit). To specify that a registry entry should be installed to a 64-bit location, add the entry to the SOFTWARE (64-Bit) node, or a subnode.

At build time, if any of the files, folders, or registry entries in the project are configured to be installed to a 64-bit location, InstallShield builds a 64-bit (x64) .msi package, which can install to 64-bit locations on 64-bit systems.

Note that a 64-bit Windows Installer–based installation can install to 64-bit locations only on 64-bit systems; they cannot be run on 32-bit systems. Note also that a 32-bit Windows Installer–based installation can be run on 64-bit systems, but it cannot install to 64-bit locations. If your product targets both 32-bit systems and 64-bit systems, you can use the Express edition of InstallShield to create one project that targets 32-bit systems, and a separate project that targets 64-bit systems.

For more information, see:

- Challenges of Supporting Both 32-Bit and 64-Bit Target Operating Systems
- Adding Files and Folders to a Project
- Displaying Predefined Folders in the Files View
- Editing the Registry
- Destination Folders

Support for Preventing a Shortcut from Being Pinned to the Windows 8 Start Screen

InstallShield lets you specify whether you want each shortcut in your installation to be pinned by default to the Start screen on Windows 8 target systems. You may want to disable pinning for shortcuts that are for tools and secondary products that are part of your installation. If you disable pinning for a shortcut, the shortcut is still available in the Apps list that contains shortcuts to all of the applications on the system.

To prevent Start Screen pinning for a shortcut, use the new Pin to Windows 8 Start Screen setting for a shortcut in the Shortcuts/Folders view.

To learn more, see Shortcuts/Folders View.

What's New in InstallShield 2012 Spring SP1 Express Edition

InstallShield 2012 Spring Service Pack 1 (SP1) includes changes that offer support for the final released versions of Windows 8, Windows Server 2012, and Visual Studio 2012. It also includes additional changes.

Support for Visual Studio 2012, .NET Framework 4.5, and Visual C++ 2012

InstallShield includes changes that offer support for the final released version of Visual Studio 2012, enabling development of installations and products within this version of the Visual Studio interface.

In addition, InstallShield includes two updated InstallShield prerequisites for the .NET Framework and two new InstallShield prerequisites for Visual C++:

- Microsoft .NET Framework 4.5 Full
- Microsoft .NET Framework 4.5 Web
- Microsoft Visual C++ 2012 Redistributable Package (x86)
- Microsoft Visual C++ 2012 Redistributable Package (x64)

The Web prerequisite for the .NET Framework requires an Internet connection. This prerequisite downloads the required redistributable files if appropriate. The full prerequisite is a stand-alone installation that does not require an Internet connection.

Additional Changes

For a list of issues that are resolved in InstallShield 2012 Spring SP1, see the release notes. The release notes are available from the Help menu in InstallShield.

What's New in InstallShield 2012 Spring Express Edition

New Features

InstallShield includes the following new features.

Ability to Target Windows 8 and Windows Server 2012 Systems

InstallShield enables you to specify that your installation requires Windows 8 or Windows Server 2012. It also lets you build feature conditions for these operating systems.

The InstallShield prerequisites that should be installable on Windows 8 and Windows Server 2012 have been updated so that they are installed on those systems if needed. Previously, the prerequisites were not run by default on those systems. This applies to the following InstallShield prerequisites:

- FSharp Redistributable Package 2.0
- Microsoft ReportViewer 2010
- Microsoft SQL CE 3.5 SP2
- Microsoft SQL Server 2005 Express SP3
- Microsoft SQL Server 2008 Express SP1

- Microsoft SQL Server 2008 Management Objects 10.00.2531
- Microsoft SQL Server 2008 Native Client 10.00.2531
- Microsoft SQL Server 2008 R2 Express RTM
- Microsoft SQL Server 2008 R2 Native Client 10.50.1600.1
- Microsoft SQL Server Native Client 9.00.4035
- Microsoft SQL Server System CLR Types 10.00.2531
- Microsoft Visual C++ 2005 SP1 Redistributable MFC Security Update KB2538242
- Microsoft Visual C++ 2005 SP1 Redistributable Package
- Microsoft Visual C++ 2008 SP1 Redistributable MFC Security Update KB2538243
- Microsoft Visual C++ 2008 SP1 Redistributable Package
- Microsoft Visual C++ 2010 Redistributable Package
- Microsoft Visual C++ 2010 RTM Redistributable MFC Security Update KB2467173
- Microsoft Visual C++ 2010 SP1 Redistributable Package
- Microsoft VSTO 2010 Runtime

Beta Support for Microsoft Visual Studio 2012

InstallShield includes support for the beta of Visual Studio 2012. You can create InstallShield projects from within this version of Visual Studio.

Microsoft .NET Framework 4.5 Prerequisites

InstallShield includes two new .NET-related InstallShield prerequisites that you can add to Express projects:

- Microsoft .NET Framework 4.5 Full
- Microsoft .NET Framework 4.5 Web

These InstallShield prerequisites install the beta versions of the .NET Framework 4.5 on supported target systems.

The Web prerequisite requires an Internet connection. This prerequisite downloads the required redistributable files if appropriate. The full prerequisite is a stand-alone installation that does not require an Internet connection.

Microsoft SQL Server 2012 Prerequisites

InstallShield includes several new SQL Server 2012–related InstallShield prerequisites that you can add to Express projects:

- Microsoft SQL Server 2012 Express
- Microsoft SQL Server 2012 Express LocalDB
- Microsoft SQL Server 2012 Native Client

InstallShield also includes InstallShield prerequisites that install Microsoft .NET Framework 3.5 SP1 Update KB956250, which is a dependency of Microsoft SQL Server 2012 Express.

These InstallShield prerequisites install the technology on supported target systems.

New InstallShield Prerequisites for SQL Server Compact 4.0 and JRE SE 1.7

InstallShield includes new InstallShield prerequisites that you can add to Express projects:

- Java Runtime Environment Second Edition (JRE SE) 1.7
- SQL Server Compact 4.0

These InstallShield prerequisites install the technology on supported target systems.

New FlexNet Connect 13.03 Redistributables Available

InstallShield includes support for FlexNet Connect 13.03 in Express projects. Use the Update Notifications view in InstallShield to include one of the two FlexNet Connect 13.03 merge modules—one has the Common Software Manager, and the other does not.

What's New in InstallShield 2012 SP1 Express Edition

Enhancements

InstallShield includes the following enhancement.

Support for Digitally Signing Software Identification Tags

If you configure your project to include a software identification tag and you also configure the release in the Releases view to use a .pfx file to digitally sign your release, InstallShield digitally signs the tag at build time. Note that the .NET Framework 2.0 or later must be installed on your build machine in order to sign a tag file.

For more information, see Including a Software Identification Tag for Your Product.

What's New in InstallShield 2012 Express Edition

New Features

InstallShield includes the following new features.

New InstallShield Prerequisites for Internet Explorer 9, SQL Server 2008 R2 Native Client, Windows Identity Foundation, and Other Redistributables

InstallShield includes several new InstallShield prerequisites that you can add to projects:

- Internet Explorer 9
- Microsoft SQL Server 2008 R2 Native Client 10.50.1600.1
- Windows Identity Foundation
- Microsoft VSTO 2010 Runtime (x64)
- Microsoft Office 2010 PIA (This prerequisite installs the Microsoft Office 2010 Primary Interop Assemblies. To use this prerequisite, download the PrimaryInteropAssembly.exe file from Microsoft's Web site and run it to extract the .msi file.)

Improvements for COM Extraction

InstallShield supports a new monitoring method for COM extraction. If you are using InstallShield on a Windows Vista or later system, this new method is used by default. The method uses a kernel driver to monitor the areas of the registry that are modified during dynamic COM extraction at build time and static COM extraction at design time. It combines the advantages that the earlier methods provided, allowing the DLL to read existing registries entries and preventing changes to the build machine.

If necessary, you can switch between the three different COM extraction methods by setting the value data of the UseAPIRegistryHooks registry value, which is in the registry key HKEY_LOCAL_MACHINE\SOFTWARE\InstallShield\RegSpy (on 32-bit machines) or HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\InstallShield\RegSpy (on 64-bit machines). Possible REG_DWORD value data are:

- **0**—Use API hooking to read existing registry entries for the DLL.
- **1**—Use registry redirection to prevent making changes to the registered DLLs on the build machine. If the value is not set, this is the default behavior on Windows XP and Windows Server 2003 systems.
- **2**—Use the new kernel mode monitoring, which combines the advantages of both of the other methods. If the value is not set, this is the default behavior on Windows Vista and later systems.

Predefined System Searches for Adobe Reader 10, Internet Explorer 9, and Microsoft Office

InstallShield has new predefined system searches:

- Adobe Reader 10
- Internet Explorer 9
- Microsoft Office 2010
- Microsoft Office 2007
- Microsoft Office 2003

If your installation requires one or more of these, you can use the Requirements view or the Installation Requirements page in the Project Assistant to add these system searches to your project. When end users launch your installation, Windows Installer checks the target system to see if the requirements are met; if they are not met, the installation displays the error message that is defined for the system search.

Support for Software Identification Tagging

ISO/IEC 19770-2 is an international standard for the creation of software identification tags. A software identification tag is an XML-based file that contains descriptive information about the software, such as the product name, product edition, product version, and publisher. Software asset management tools collect the data in the tags to provide accurate application identification for software that is installed in an enterprise.

Software identification tagging is evolving as an industry standard, enabling independent software vendors to create smarter applications that give their customers better information for software asset management and license optimization initiatives. Including the identification tag in your product's installation makes it possible for your customers to use tools that can monitor their internal usage of your product, allowing them to manage and optimize the number of licenses of your product that they obtain from you, and stay in compliance with your licensing policies.

InstallShield includes several new settings in the General Information view that let you specify information that is required to create an identification tag for your product. This view also contains a new Use Software Identification Tag setting that lets you specify whether you want InstallShield to create the tag at build time and include it in your installation; the default value of this setting is Yes.

Note that if Yes is selected for the Use Software Identification Tag setting but you have not entered values in one or more of the required identification settings (the Unique ID, Tag Creator, and Tag Creator ID settings in the General Information view), build warning -7235 occurs, once for each of the settings that are blank. This build warning explains that the software identification tag could not be created and included in the installation because a specific required setting was left blank. To resolve this warning, enter appropriate value in each specific setting, or select No for the Use Software Identification Tag setting.

This feature applies to Express projects.

For more information, see:

- Including a Software Identification Tag for Your Product
- General Information Settings

What's New in InstallShield 2011 Express Edition

New Features

InstallShield includes the following new features.

Integration with Team Foundation Server (TFS)

InstallShield has enhanced support for integrating with Team Foundation Server (TFS) 2010.

Now when you are using InstallShield from within Visual Studio 2010, you can access the Source Control Explorer to integrate your InstallShield project with Team Foundation version control and manage changes to your InstallShield projects and your Visual Studio solutions.

You can also use Team Foundation Build to compile, test, and deploy your InstallShield projects and your Visual Studio solutions on a regular basis, or on demand. Your installation is automatically updated with your latest source files every time that your solution is built.

In addition, if you install Team Explorer on the same machine that has InstallShield and Visual Studio, you can use Team Explorer from within your InstallShield projects that are open in Visual Studio. This enables you to perform tasks such as the following:

- Use Source Control Explorer when you are working on your InstallShield projects.
- Configure builds for your InstallShield projects and Visual Studio solutions.
- Queue new builds.
- Track work items such as bugs and tasks for your InstallShield projects and your Visual Studio solutions.

For more information, see:

- Integrating with Microsoft Visual Studio Team Foundation Server
- Adding InstallShield Projects to Team Explorer

New InstallShield Prerequisites for SQL Server 2008 R2 Express, SQL Server Native Client, Visual C++ 2010, and Other Redistributables

InstallShield includes a number of new InstallShield prerequisites that you can add to Express projects:

- Microsoft SQL Server 2008 R2 Express
- Microsoft SQL Server 2008 Native Client 10.00.2531
- Microsoft SQL Server Native Client 9.00.4035
- Microsoft SQL Server System CLR Types 10.00.2531
- Microsoft SQL Server 2008 Management Objects 10.00.2531
- Microsoft Visual C++ 2010 Redistributable Package
- Microsoft Visual C++ 2008 SP1 Redistributable Package
- Windows Installer 3.1 - Japanese
- MSXML 6.0 SP1 - Japanese
- Microsoft .NET Framework 4.0 Client Japanese Language Pack
- Microsoft .NET Framework 4.0 Full Japanese Language Pack

Ability to Specify a Search Path for InstallShield Prerequisites

InstallShield now lets you specify the folders where InstallShield should search for InstallShield prerequisite files (.prq files). This functionality makes it easier for teams of users to share InstallShield prerequisites with each other, and for storing prerequisites in source code control. Previously, InstallShield searched for .prq files in the following location only: *InstallShield Program Files Folder*\SetupPrerequisites.

InstallShield offers several ways for specifying the folders:

- If you are editing or building from within InstallShield, use the new Prerequisites tab on the Options dialog box—which is displayed when you click Options on the Tools menu—to specify a comma-delimited list of machine-wide folders and current-user folders. This tab is similar to the Merge Modules tab on the Options dialog box, which lets you specify search paths for merge modules.
- If you are building from the command line with `ISCmdBld.exe`, use the new `-prqpath` parameter to specify a comma-delimited list of folders.

  If you use an .ini file to specify `ISCmdBld.exe` parameters, you can use the new PrerequisitePath parameter in the [Mode] section of your .ini file to specify a comma-delimited list of folders.
- If you are building through MSBuild or Team Foundation Server (TFS), use the new PrerequisitePath parameter on the InstallShield task. This parameter is exposed as the ItemGroup InstallShieldPrerequisitePath when the default targets file is used. To specify multiple paths, use an ordered array of paths.

To learn more, see:

- Specifying the Directories that Contain InstallShield Prerequisites
- Prerequisites Tab
- IsCmdBld.exe
- Passing Command-Line Build Parameters in an .ini File

- Microsoft Build Engine (MSBuild)

Ability to Specify Custom Version Resource Properties for Setup.exe and Update.exe

InstallShield now lets you use a custom version resource properties for `Setup.exe` files that you create at build time. The version resource properties are displayed on the Properties dialog box for `Setup.exe`; this Properties dialog box opens when end users right-click the `Setup.exe` file and then click Properties. This support is available in Express projects. The ability to specify custom version resource properties is also now available for `Update.exe` files that you create in QuickPatch projects.

Two new settings—Launcher Copyright and File Description—are available on the Setup.exe tab for a release in the Releases view. These settings let you specify a custom copyright notice and file description.

When InstallShield is configuring the following version resources of your `Setup.exe` setup launcher at build time, it now uses the custom information that you entered in the General Information view and the Releases view:

- Company name
- Product name
- Product version
- Copyright
- File version
- File description

Previously, InstallShield did not use any custom information in many cases. For example, the `Setup.exe` files that InstallShield previously created contained details that pertained to the version of InstallShield that was used to build the `Setup.exe` file. Thus, the copyright notice that was displayed in the `Setup.exe` Properties dialog box was the copyright notice for InstallShield, rather than the copyright notice for your product.

Several new settings—Company Name, Product Name, Product Version, Description, and Copyright—let you specify the custom information that you want InstallShield to use when you build an `Update.exe` file. These settings are available on the Advanced tab of the Build Settings area in the General Information view of QuickPatch projects.

Previously, InstallShield did not use any custom version resource information for `Update.exe` files.

To learn more, see:

- Customizing File Properties for the Setup Launcher
- Setup.exe Tab
- Customizing File Properties for the Update Launcher
- Advanced Tab

Improvements for Custom Actions: New Custom Action Types; New Maintenance Mode Sequence Options; Support for Rollback and Commit Custom Actions

InstallShield includes several new features for custom action support.

MSI DLL Custom Actions

InstallShield now lets you add MSI DLL custom actions to your projects. An MSI DLL custom action calls an entry-point function that is exported from a C or C++ DLL. The DLL can be installed at run time during the installation, or it can be embedded within the `Binary` table of the .msi package. The MSI DLL custom action offers more flexibility for the DLL file's source location than the standard DLL custom action, which was the only type of DLL custom action that was supported in earlier versions of InstallShield. During some parts of the installation, standard DLL actions cannot be stored in the `Binary` table; instead, they must be installed with the product. However, during those same parts of the installation, MSI DLL actions can be stored in the `Binary` table or installed with the product.

Note that for MSI DLL custom actions, a return value of zero indicates success; non-zero return values indicate statuses such as failure or cancellation. This is consistent with all other custom action types, except for standard DLL custom actions. For standard DLL custom actions, a non-zero return value indicates success, and a return value of zero indicates failure.

JScript Custom Actions

InstallShield also lets you add JScript custom actions to your projects. If you convert a Visual Studio setup project (.vdproj) that has a JScript custom action into an Express project (.ise), InstallShield now includes the JScript custom action.

Separate Maintenance and Uninstallation Sequences

The Custom Actions view contains a new Custom Actions During Maintenance node that lets you schedule actions that you want to occur only during maintenance. The following sequence nodes have been moved from the Custom Actions During Uninstallation node to this new Custom Actions During Maintenance node:

- After Initialization (before first dialog)
- After Maintenance Welcome dialog
- After Maintenance Type dialog
- After Ready to Remove dialog
- After Setup Progress dialog
- After Setup Complete Success dialog

The following two sequences are still applicable to uninstallation, so they remain under the Custom Actions During Uninstallation node: the Before System Changes sequence and the After System Changes sequence. Any custom actions that are scheduled during these sequences are run during uninstallation, but not during maintenance operations such as repair.

Rollback Actions, Commit Actions, and Related Settings

The Custom Actions view has a new In-Script Execution setting that lets you select which iteration of the execute sequence—deferred, rollback, or commit—should trigger your custom action. Deferred actions make the run-time changes to the system. Rollback actions occur if the installation encounters an error or the end user cancels the installation before it completes; these actions are intended to undo the changes to the system. Commit actions perform cleanup for any temporary information that is saved by a deferred action. The In-Script Execution setting also lets you specify whether your action should be run in the user context (using the privileges of the user running the installation) or in the system context (with elevated privileges).

This new In-Script Execution setting is displayed in the Custom Actions view when you select a custom action that is not sequenced to run in immediate execution mode; actions that run in immediate-execution mode can set Windows Installer properties and check the target system, and they are always run in the user context.

Previously in the Express edition of InstallShield, any custom actions that were scheduled for deferred sequences of the installation were launched in deferred-execution mode in the user context. InstallShield did not have support for rollback or commit custom actions, and it did not have support for running custom actions in the system context.

For more information, see:

- Using Custom Actions
- Windows Installer DLL Custom Actions
- VBScript and JScript Custom Actions
- Action Execution Options

Ability to Import Visual Studio Setup and Merge Module Projects into Existing InstallShield Projects; Improvements for the Project Converter

InstallShield now lets you import a Visual Studio setup project or a Visual Studio merge module project (.vdproj) into an Express project (.ise). This functionality enables you to develop InstallShield installation projects that contain the same data and settings that were in your Visual Studio project. The wizard imports the project outputs, files, registry keys, file extensions, custom actions, target system searches, and launch conditions from your Visual Studio project into your existing InstallShield project.

To import a Visual Studio project into an existing InstallShield project, use the new Visual Studio Deployment Project Import Wizard in InstallShield. The wizard lets you choose whether to import or ignore certain settings in the Visual Studio project.

The existing support for converting a Visual Studio project into a new InstallShield project has been expanded. If your Visual Studio project contains predefined prerequisites, InstallShield now converts them to equivalent InstallShield prerequisites during the conversion process. This same prerequisite conversion functionality is available if you use the new wizard to import a Visual Studio project into an InstallShield project.

If your Visual Studio project contains one or more project outputs, use the import wizard instead of the conversion process. The InstallShield project must be in the same Visual Studio solution that contains the Visual Studio setup or merge module project and all of its project dependencies. Note that you must be using InstallShield from within Visual Studio when you use the import wizard in order for the wizard to import the project outputs into your InstallShield project.

To learn more, see the following:

- Converting or Importing Visual Studio Projects into InstallShield Projects
- Visual Studio Deployment Project Import Wizard

Predefined System Searches for SQL Server 2008 Express SP1 and Adobe Reader 9

InstallShield has new predefined system searches:

- SQL Server 2008 Express SP1
- Adobe Reader 9

If your installation requires one or both of these, you can use the Requirements view or the Installation Requirements page in the Project Assistant to add these system searches to your project. When end users launch your installation, Windows Installer checks the target system to see if the requirements are met; if they are not met, the installation displays the error message that is defined for the system search.

This feature applies to Express projects.

Ability to Configure MIME Types for IIS Web Sites, Applications, and Virtual Directories

The Internet Information Services view has a new MIME Types setting that you can configure for a Web site, application, or virtual directory in your project. This setting lets you identify the types of content that can be served from the Web server on the target system to a browser or mail client.

This feature applies to the Express project.

For more information, see:

- MIME Types setting for a Web site
- MIME Types setting for an application or virtual directory

Support for Looping Through Image Billboards

The Billboards view has a new Loop Billboards setting that lets you specify whether you want your installation to continuously loop the image billboards until the file transfer completes and the installation displays the appropriate Setup Complete dialog.

If you select Yes for this setting and the file transfer takes more time than you have allocated for the billboards, the installation restarts the display of billboards from the beginning. The loop continues, if necessary, until the file transfer ends. The default value for this setting is No, which matches the behavior in earlier versions of InstallShield.

Previously, if the file transfer took more time than what you had allocated for the billboards, the installation continued to display the last billboard until the file transfer finished; it did not loop the billboards.

This feature applies to Express projects.

For more information, see:

- Run-Time Behavior of an Installation that Includes Billboards
- Billboard Settings

New FlexNet Connect 12.01 Redistributables Available

InstallShield includes support for FlexNet Connect 12.01 in InstallShield projects. Use the Update Notifications view in InstallShield to include one of the two FlexNet Connect 12.01 merge modules—one has the Common Software Manager, and the other does not.

Enhancements

InstallShield includes the following enhancements.

Unicode Views in InstallShield

Some of the views InstallShield have been enhanced to display and allow you to enter characters from all languages. For example, now you can use Chinese characters in system software requirement messages if you are configuring software requirements in the Requirements view on an English machine. Previously, the characters in the messages were displayed as question marks.

The areas in InstallShield that have been enhanced to include the Unicode support are the Requirements view and the tabs for a release in the Releases view. Note that Unicode support was previously introduced in many other views in InstallShield 2010 Express Edition.

The improvements are applicable to Express projects.

Support for Adding Project Outputs from Visual Studio Web Setup Projects

If you create a Visual Studio solution that includes a Web setup project and an InstallShield installation project and you are using InstallShield from within Visual Studio, you can now add project outputs from the Web setup project to your InstallShield project.

Ability to Specify the Required Execution Level for Update.exe Manifests

The Required Execution Level setting is now available on the Advanced tab of the Build Settings area in the General Information view of QuickPatch projects. Use this new setting to specify the minimum execution level that is required by your project's `Update.exe` file for running the upgrade on Windows Vista and later platforms. InstallShield adds a manifest that specifies the required level. By default, InstallShield uses the level that was configured in the previous setup launcher's manifest.

Previously, the Required Execution Level setting was available only for the `Setup.exe` setup launchers. If you created an `Update.exe` patch, InstallShield used the same required execution level that was configured in the previous setup launcher's manifest.

To learn more, see Advanced Tab.

Ability to Create .cab Files Without Compression

A new Cab Optimization Type setting is available on the Build tab for a release that is selected in the Releases view. If Compressed or one of the custom options is selected for the Compression setting, use the Cab Optimization Type setting to specify the type of compression that InstallShield should use when building the release's .cab files. The available options are LZX compression, MSZIP compression, or no compression.

The Cab Optimization Type setting replaces the Optimize Size setting, which had support for only LZX compression and MSZIP compression; it did not let you skip compression.

This enhancement is available in Express projects.

For more information, see Build Tab.

New Refresh Button in the Redistributables and Prerequisites Views

The Redistributables view contains a new Refresh button that you can use to refresh the list of redistributables that are displayed in the view. Previously, if you added redistributables to your machine when this view was open in InstallShield, it was necessary to close and then reopen the project in order to see an updated list in the view.

Important Information

Installing More than One Edition of InstallShield

Only one edition of InstallShield 2011—Premier, Professional, or Express—can be installed on a system at a time.

Installing More than One Version of InstallShield

InstallShield 2026 Express Edition can coexist on the same machine with other versions of InstallShield.

What's New in InstallShield 2010 Express Edition

Expansion Pack for Visual Studio 2010

InstallShield 2010 Express Edition Expansion Pack for Visual Studio 2010 includes changes that offer support for the final released version of Visual Studio 2010 and .NET Framework 4. It also includes additional changes.

Microsoft .NET Framework 4 Prerequisites

InstallShield includes four new .NET-related InstallShield prerequisites that you can add to Express projects:

- Microsoft .NET Framework 4.0 Full
- Microsoft .NET Framework 4.0 Full (Web Download)
- Microsoft .NET Framework 4.0 Client
- Microsoft .NET Framework 4.0 Client (Web Download)

Note the following details about these prerequisites:

- The full prerequisites install the .NET Framework runtime and associated files that are required to run and develop applications that target the .NET Framework 4.
- The client prerequisites install the .NET Framework runtime and associated files that are required to run most client applications.
- The two Web download prerequisites require an Internet connection. These prerequisites download the required redistributable files if appropriate. The other two prerequisites are stand-alone installations that do not require an Internet connection.

The .NET Framework 4.0 requires Windows Installer 3.1 or later, as well as the Windows Imaging Component. Therefore, the .NET Framework 4.0 Full prerequisites are configured to have dependencies for the following InstallShield prerequisites:

- Windows Installer 3.1 (x86)
- Windows Imaging Component (x86)
- Windows Installer 3.1 for Windows Server 2003 SP1 (x86)
- Windows Imaging Component (x64)
- Windows Installer 3.1 for Windows Server 2003 SP1 (IA64)
- Windows Installer 3.1 for Windows Server 2003 SP1 (x64)

- Windows Installer 3.1 for Windows XP (x64)

In addition, the .NET Framework 4.0 Client prerequisites are configured to have dependencies for the following InstallShield prerequisites:

- Windows Installer 3.1 (x86)
- Windows Imaging Component (x86)
- Windows Installer 3.1 for Windows Server 2003 SP1 (x86)
- Windows Imaging Component (x64)
- Windows Installer 3.1 for Windows Server 2003 SP1 (x64)
- Windows Installer 3.1 for Windows XP (x64)

Thus, if you add any of the .NET Framework 4.0 prerequisites to your project, InstallShield also adds prerequisites for Windows Installer and Windows Imaging Component to your installation automatically by default. This could increase the size of your installation.

Additional Prerequisites for Visual Studio 2010 Support

InstallShield includes the following new InstallShield prerequisites that you can add to Express projects:

- Microsoft SQL CE 3.5 SP2
- Microsoft ReportViewer 2010
- Microsoft VSTO 2010 Runtime
- FSharp Redistributable Package 2.0
- Microsoft Office 2007 PIA (This prerequisite installs the Microsoft Office 2007 Primary Interop Assemblies. To use this prerequisite, download the `PrimaryInteropAssembly.exe` file from Microsoft's Web site and run it to extract the `o2007pia.msi` file. Note that you may need to change the path of the `o2007pia.msi` installation in the .prq file, depending on where the .msi package is located on your system.)

Predefined System Searches for .NET Framework 4

InstallShield includes two new predefined system searches:

- Microsoft .NET Framework 4.0 Full package
- Microsoft .NET Framework 4.0 Client package

If your installation requires either of these, you can use the Installation Requirements page in the Project Assistant or the Requirements view to add these system searches to your project. When end users launch your installation, Windows Installer checks the target system to see if the requirements are met; if they are not met, the installation displays the error message that is defined for the system search.

This functionality applies to Express projects.

What's New in InstallShield 2010 Express Edition SP1

InstallShield 2010 Express Edition Service Pack 1 (SP1) includes changes that offer support for the final released versions of Windows 7, Windows Server 2008 R2, and Windows Installer 4.5. It also includes additional changes.

Important ▪ *If you want to open an InstallShield 2010 project in InstallShield 2010 SP1, you must allow InstallShield to upgrade your project to InstallShield 2010 SP1. InstallShield 2010 SP1 includes support for tables that were not available in InstallShield 2010 projects, and these tables need to be added during the upgrade. Note that it is not possible to open InstallShield 2010 SP1 projects in earlier versions of InstallShield (including InstallShield 2010 without SP1). Therefore, if multiple users need to open and modify your InstallShield projects, ensure that all of you apply the SP1 patch at the same time.*

If you open an InstallShield 2010 project in InstallShield 2010 SP1, InstallShield 2010 SP1 displays a message box that asks you if you want to convert the project to the new version. If you reply that you do want to convert it, InstallShield creates a backup copy of the project before converting it.

Setup.exe Manifests Now Have Compatibility Section to Avoid Triggering Program Compatibility Assistant on Windows 7 and Windows Server 2008 R2 Systems

If you configure your InstallShield project to create a setup launcher for your installation, the manifest that InstallShield creates for the setup launcher now includes a compatibility section. Previously, without this compatibility section, a Program Compatibility Assistant (PCA) dialog box may have appeared at the end of an installation on Windows 7 and Windows Server 2008 R2 systems. The PCA dialog box indicated that the program might not have installed correctly. This dialog box was displayed if the installation did not create the application uninstallation key. This may happen if the end user cancels the installation or the installation fails to complete successfully.

Additional Changes

For a list of issues that are resolved in InstallShield 2010 Express Edition SP1, see the release notes. The release notes are available from the Help menu in InstallShield.

What's New in InstallShield 2010 Express Edition

New Features

InstallShield 2010 Express Edition includes the following new features.

Ability to Target Windows 7 and Windows Server 2008 R2 Systems

InstallShield enables you to specify that your installation requires Windows 7 or Windows Server 2008 R2. It also lets you build feature and custom action conditions for these operating systems.

Windows 7 and Windows Server 2008 R2 Support for Displaying Installation Progress on the Taskbar

Installations that are run on Windows 7 and Windows Server 2008 R2 now show a progress bar on the Windows taskbar during file transfer. This applies to installations that display billboards that were configured in the Billboards view. Note that a progress bar is not displayed on the taskbar on earlier versions of Windows. It is also not displayed during setup initialization or while InstallShield prerequisites are being installed.

Beta Windows Installer 5 Support for Per-User Installations

The Show All Users Option setting in the Dialogs view now has support for the `MSIINSTALLPERUSER` property that is available with the beta of Windows Installer 5. Use this setting to specify whether you want to give end users the option of installing your product for all users or for only the current user. Depending on the value that you select

for this setting, the Ready to Install dialog may include buttons that let end users specify how they want to install the product; the buttons are displayed if the installation is run on a system that has Windows 7 or Windows Server 2008 R2.

Note that the Show All Users Option setting is now available when you select the main Dialogs node in the Dialogs view. Previously, this setting was available if you selected the Customer Information dialog in this view.

This feature is available in Express projects.

For detailed information, see:

- Per-User vs. Per-Machine Installations
- Global Dialog Settings for All End-User Dialogs

Beta Windows Installer 5 Support for Reducing the Time for Installing Large Packages

Use the new Fast Install setting in the General Information view to select one or more options that may help reduce the time that is required to install a large Windows Installer package. For example, you can specify that you do not want a system restore point to be saved for your installation. You can also specify that you want the installation to perform file costing, but not any other costing.

This setting configures the new Windows Installer property `MSIFASTINSTALL`, which can be set at the command line. Windows Installer 5 includes support for this property. Earlier versions of Windows Installer ignore it.

This setting is available in Express projects.

To learn more, see the description of the Fast Install setting.

New Support for Setting Permissions for Files, Folders, and Registry Keys

InstallShield offers a new way to secure files, folders, and registry keys for end users who run your product in a locked-down environment: With the new custom InstallShield handling method, InstallShield stores permission information for your product in the custom `ISLockPermissions` table of the .msi database. InstallShield also adds custom actions to your project to set the permissions. This support is available in Express projects.

Previously, the only option that InstallShield offered for setting permissions was to use the traditional Windows Installer handling. With this option, the permission information is stored in the `LockPermissions` table of the .msi database. The new custom InstallShield handling option offers several advantages over the traditional Windows Installer handling:

- The custom option includes support for many well-known security identifiers (SIDs) that are not supported by the traditional Windows Installer handling option.
- The custom option supports the use of localized user names for the supported SIDs, unlike the traditional option. With the traditional option, if you try to use a localized name to set permissions on a non-English system, the installation may fail.
- The custom option lets you specify that you want to deny a user or group from having the permissions that you are specifying. The traditional handling does not allow you to do this.
- The custom option lets you add permissions to a file, folder, or registry key that already exists on the target system, without deleting any existing permissions for that object. With the traditional handling, the existing permissions are deleted.
- The custom option lets you configure permissions for a folder (or a registry key), and indicate whether you want the permissions to be applied to all of the folder's subfolders and files (or the registry key's subkeys).

With the traditional handling, if you want to configure permissions for a subfolder or a file in a folder (or a subkey under a registry key), the parent that is created on the target system automatically inherits the permissions of its child.

- The custom option lets you configure permissions for a new user that is being created during the installation. The traditional handling does not allow you to do this; the user must already exist on the target system at run time.

The General Information view has a new Locked-Down Permissions setting that lets you specify whether you want to use the new custom InstallShield handling or the traditional Windows Installer handling for all new permissions that you set for files, folders, and registry keys in your project. If you have already configured some permissions in your project and then you change the value of this setting, InstallShield lets you specify whether you want to use the alternate handling method for those already-existing permissions. In all new projects, the default value for this setting is the custom InstallShield handling option. If you upgrade a project from InstallShield 2009 Express Edition or earlier to InstallShield 2010 Express Edition, the traditional Windows Installer handling option is the default value of this setting. This new setting is available in Express projects.

For more information, see the following:

- Securing Files, Folders, and Registry Keys in a Locked-Down Environment
- Selecting the Locked-Down Permissions Type for a Project

New InstallShield Prerequisites for Windows Installer, .NET Framework, Crystal Reports, and Other Redistributables

InstallShield includes a number of new InstallShield prerequisites that you can add to Express projects:

- Windows Installer 4.5 (The InstallShield prerequisites for Windows Installer 4.5 include the fix that is described in Microsoft KB958655.)
- Windows Installer 4.5 Update (The InstallShield prerequisites for the Windows Installer 4.5 Update include the fix that is described in Microsoft KB958655. Windows Installer 4.5 must already be installed on the target system for this update.)
- Windows Installer 3.1, Windows Installer 3.0, and Windows Installer 2.0 (These versions of Windows Installer redistributables were previously available if you added Windows Installer to your project in the Releases view. These versions were not previously available as InstallShield prerequisites.)
- .NET Framework 3.0 SP1
- .NET Framework 2.0 SP2
- Internet Explorer 8
- Microsoft SQL Server 2008 Express SP1
- Microsoft SQL Server 2005 Express SP3
- Microsoft Visual C++ 2005 SP1 Redistributable Package
- Crystal Reports Basic for Visual Studio 2008 (Use this prerequisite with the Crystal Reports Basic installation that is installed with Visual Studio 2008. Note that you may need to change the path of the Crystal Reports Basic installation in the .prq file, depending on where the .msi package is located on your system.)

Ability to Add IIS Web Applications to Web Sites Without Virtual Directories

InstallShield now lets you add IIS Web applications to Web sites. You can do so by right-clicking a Web site in the Internet Information Services view and clicking New Application. Once you have added a new application, you can configure its settings in the right pane.

InstallShield also lets you create a virtual directory without an application. Previously whenever you created a virtual directory, an application was also created automatically.

This feature applies to Express projects.

To learn more, see:

- Creating a Web Site and Adding an Application or a Virtual Directory
- Internet Information Services View

New FlexNet Connect 11.6 Redistributables Available

InstallShield includes support for FlexNet Connect 11.6 in Express projects. Use the Update Notifications view in InstallShield to include one of the two FlexNet Connect 11.6 merge modules—one has the Common Software Manager, and the other does not. These merge modules replace the FlexNet Connect 11 merge modules.

Enhancements

InstallShield 2010 Express Edition includes the following enhancements.

Usability Enhancements

Some of the views in InstallShield have been enhanced to improve productivity and usability. For example, several views contain new toolbars that make options easier to find. Some of the views that contain grids let you customize how the rows in a grid are organized. Searches are performed more quickly in the views that offer search capabilities. Following are examples of some of the highlights:

- Redistributables view—The new toolbar and the new group box area in this view provide robust search and organizational functionality. You can drag and drop column headings onto the group box area to organize the list of redistributables in a hierarchical format. In addition, you can type a string in the toolbar's search box, and InstallShield hides all of the redistributables that do not contain it.
- Internet Information Services view—This view has been redesigned to look similar to IIS 7: the settings are now displayed in grids, instead of on tabs. The grids have buttons that let you sort the grid settings by category or alphabetically. When you select a setting in one of the grids in this view, InstallShield displays help information for that setting in the lower-right pane.
- General Information view—The settings in this view are grouped into several categories in a grid to make it easy to find a particular setting. The grid has a button that lets you sort the grid settings by category or alphabetically.

In addition, the Output window, which is displayed when you are building a release or upgrading a project, has been enhanced. The Output window or its individual tabs can be docked to any side of the workspace in InstallShield, or they can be dragged to free-floating positions. If you drag the Output window or one of its tabs to the edge of the InstallShield interface, it becomes a docked window. If you drag the Output window or one of its tabs away from any of the edges of the InstallShield interface, it becomes undocked.

For more information, see:

- Working with the Group Box Area in Various Views

- Docking or Undocking the Output Window

Predefined System Searches for the .NET Framework and Internet Explorer 8

InstallShield has two new predefined system searches:

- Microsoft .NET Framework 3.5 SP1
- Internet Explorer 8

If your installation requires one or both of these, you can use the Requirements view or the Installation Requirements page in the Project Assistant to add these system searches to your project. When end users launch your installation, Windows Installer checks the target system to see if the requirements are met; if they are not met, the installation displays the error message that is defined for the system search.

This enhancement applies to Express projects.

What's New in InstallShield 2009 Express Edition

New Features

InstallShield 2009 Express Edition includes the following new features.

Ability to Associate InstallShield Prerequisites with Features for Chaining Installations

InstallShield now enables you to associate InstallShield prerequisites with one or more features. This new type of InstallShield prerequisite is called a *feature prerequisite*. It is installed if a feature that contains the prerequisite is installed and if the prerequisite is not already installed on the system.

Including InstallShield prerequisites in your project enables you to chain multiple installations together, bypassing the Windows Installer limitation that permits only one Execute sequence to be run at a time. The `Setup.exe` setup launcher serves as a bootstrap application that manages the chaining.

The Redistributables view is where you add InstallShield prerequisites to a project and specify whether you want them to run before your main installation or be associated with one or more features in your main installation.

Previously, all InstallShield prerequisite installations were run before the main installation ran, and the InstallShield prerequisites could not be associated with any features. This type of prerequisite, which is still available, is called a *setup prerequisite*.

To learn more, see:

- Setup Prerequisites vs. Feature Prerequisites
- Associating an InstallShield Prerequisite with a Feature in a Project
- Run-Time Behavior for an Installation that Includes InstallShield Prerequisites
- Working with InstallShield Prerequisites that Are Included in Projects

Billboard Improvements: Support for Adobe Flash Application File (.swf) Billboards and More Image Billboard File Types (.gif, .jpg, .jpeg), New Billboard Styles, and Ability to Preview Billboards

InstallShield includes several new billboard-related features that give you more flexibility and control over the look and feel of the file transfer portion of your installation:

- You can add an Adobe Flash application file (.swf) as a billboard in your project.

  Flash application files can consist of videos, movies, sounds, interactive interfaces, games, text, and more—anything that is supported by the .swf type of file.

- InstallShield lets you use .gif, .jpg, and .jpeg files as billboards. Previously, only .bmp files were supported.

- InstallShield includes a new Billboard Type setting that lets you specify which style of billboard you want to use in your installation. For example, with one style, the installation displays a full-screen background, with billboards in the foreground, and a small progress box in the lower-right corner of the screen. With another style, the installation displays a standard-size dialog that shows the billboards. The bottom of this dialog shows the progress bar.

- InstallShield lets you preview a billboard to see how it would be displayed at run time, without requiring you to build and run a release. Previewing a billboard lets you see how your billboard will look with the background color, position, and related settings that are currently configured for your billboard.

The Billboards view in InstallShield is where you add billboard files, configure billboard-related settings, and preview billboards.

This feature applies to Express projects.

To learn more, see:

- Displaying Billboards
- Billboard File Types
- Types of Billboards
- Adding an Adobe Flash Application File Billboard
- Adding Image Billboards
- Previewing Billboards Without Building and Launching a Release
- Run-Time Behavior of an Installation that Includes Billboards
- Billboards View

Windows Installer 4.5 Redistributables

InstallShield includes several InstallShield prerequisite files (.prq) for Windows Installer 4.5:

- Windows Installer 4.5 for Windows Vista and Server 2008 (x86)
- Windows Installer 4.5 for Windows Server 2003 SP1 and later (x86)
- Windows Installer 4.5 for Windows XP SP2 and later (x86)

This feature applies to Express projects.

For more information, see Adding Windows Installer Redistributables to Projects.

Ability to Create Unicode Versions of the Setup.exe and Update.exe Bootstrappers

InstallShield now enables you to specify whether you want to create a Unicode version or an ANSI version of the `Setup.exe` setup launcher for a project. Previously, if your project included a setup launcher, InstallShield always built an ANSI version; it did not include support for building a Unicode version.

A Unicode setup launcher can correctly display double-byte characters in the user interface of the setup launcher, regardless of whether the target system is running the appropriate code page for the double-byte-character language. An ANSI setup launcher displays double-byte characters in the setup launcher dialogs if the target system is running the appropriate code page. However, it displays garbled characters instead of double-byte characters in those dialogs if the target system is not running the appropriate code page.

Use the new Setup Launcher Type setting on the Setup.exe tab for a release in the Releases view to specify whether you want to use Unicode or ANSI. Unicode is the default type for all new Express projects.

InstallShield also now enables you to specify whether you want to create a Unicode version or an ANSI version of the `Update.exe` update launcher for a QuickPatch package. Use the new Update Launcher Type setting on the Advanced tab in the Build Settings area of the General Information view to specify whether you want to use Unicode or ANSI. Unicode is the default type for all new QuickPatch projects.

Ability to Create a Log File for the Setup.exe Bootstrapper

A new `/debuglog` command-line parameter is available for the `Setup.exe` setup launcher for Express projects. Use this command-line parameter to generate a log file for debugging. For more information, see /debuglog.

New Microsoft .NET Redistributables Available

InstallShield now includes many new .NET-related InstallShield prerequisites that you can add to Express projects:

- Microsoft .NET Framework 3.5 (Web Download)
- Microsoft .NET Framework 3.5 (Full Package)
- Microsoft .NET Framework 3.5 Language Packs
- Microsoft .NET Framework 3.0 SP1 (Web Download)
- Microsoft .NET Framework 3.0 Language Packs
- Microsoft .NET Framework 2.0 SP1

You can add any of these InstallShield prerequisites to your project through the Redistributables view.

For more information, see:

- Adding .NET Framework Redistributables to Projects
- Including the Microsoft .NET Framework and Microsoft .NET Framework Language Pack Prerequisites

Ability to Compress Files that Are Streamed into Setup.exe and to Specify the Compression Level

If you build a release that uses a `Setup.exe` setup launcher, InstallShield now compresses files that it streams into the `Setup.exe` file. The default compression level that InstallShield uses offers a balance between file size and time that is required to extract the compressed files at run time. If you want to change the compression level or you do not want to use any compression, you can override the default level through a machine-wide setting.

By default, InstallShield does not compress any files that have a .cab file extension when it is streaming them into the `Setup.exe` file at build time, since .cab files are already a compressed type of file. You can modify this default compression exclusion list to include other file types or specific files as needed. The exclusion list is a machine-wide setting.

For more information, see Configuring the Compression Level for Files that Are Streamed into Setup.exe.

Support for Multi-Part .cab Files

The .cab file type has some limitations. For example, the maximum size of a single .cab file is 2 GB. In addition, some users have had trouble signing large .cab files and verifying the digital signature of large signed .cab files. To work around these limitations, InstallShield now has a new default limit of 600 MB for a .cab file. When InstallShield is creating the .cab files for your release and it reaches the limit, it splits the data into two or more .cab files, creating multi-part .cab files.

You can modify the maximum size limit if necessary. In addition, if you do not want InstallShield to create multi-part .cab files, you can configure it to create single .cab files.

This functionality applies to Express projects. In addition, it is applicable only if you are building a compressed SingleImage release in which all of the files are embedded in the single-file .msi package or the `Setup.exe` setup launcher.

For more information, see Configuring the Maximum Size for .cab Files.

Support for Installing IIS 7 Web Sites on Windows Server 2008

InstallShield now lets you create and manage IIS 7 Web sites and virtual directories on Windows Server 2008 systems. This functionality is available in Express projects.

Microsoft SQL Server 2005 Express SP2 Prerequisite Available

InstallShield now includes an InstallShield prerequisite for Microsoft SQL Server 2005 Express Edition SP2. You can add this InstallShield prerequisite to Express projects.

Microsoft Visual Studio 2008 Support

InstallShield is now integrated with Visual Studio 2008, enabling development of installations and products within the same Visual Studio interface.

Ability to Convert Visual Studio Setup Projects to InstallShield Projects

InstallShield now lets you convert a Visual Studio 2008, Visual Studio 2005, Visual Studio .NET 2003, or Visual Studio .NET setup project (.vdproj) to an Express project (.ise).

To learn more, see Converting or Importing Visual Studio Projects into InstallShield Projects.

New FlexNet Connect 11 Redistributables Available

InstallShield includes support for FlexNet Connect 11. Use the Update Notifications view in InstallShield to include one of the two FlexNet Connect 11 merge modules—one has the Common Software Manager, and the other does not.

Enhancements

InstallShield 2009 Express Edition includes the following enhancements.

Best Practice Dynamic File Linking

When you add or modify a dynamic file link in your project, you can now specify which component creation method you want InstallShield to use: a new best practice method, or the previously available one-component-per-directory method.

A component is the smallest installable part of a product. The Express edition of InstallShield creates components for you automatically.

When best practices for component creation are followed, InstallShield creates a separate component for each portable executable (PE) file in the dynamically linked folder and sets each PE file as the key file of its component. This method of component creation, in combination with the streamlined QuickPatch package functionality, enables you to create patches according to Windows Installer component rules.

Previously, any time that you added dynamic file links to a project, InstallShield automatically created one component for all of the dynamically linked files at build time. However, if your dynamic file link included PE files, Windows Installer best practices for component creation were not followed.

By default, InstallShield considers the following file types to be PE files: .exe, .dll, .ocx, .vxd, .chm, .hlp, .tlb, and .ax. You can modify this list through the File Extensions tab on the Options dialog box.

The best practice dynamic file linking applies to Express projects.

To learn more, see:

- Determining the Appropriate Component Creation Method for Dynamically Linked Files
- File Extensions Tab

Ability to Install IIS Web Sites Without Virtual Directories

InstallShield now includes support for installing IIS Web sites without any virtual directories. This support is available for all new Web sites that are created in InstallShield.

Previously, InstallShield did not include support for installing Web sites without virtual directories. Therefore, if a Web site in an installation did not have any virtual directories, the Web site would not be created at run time.

Note that if you upgrade an InstallShield 2008 or earlier project to InstallShield 2026 Express Edition, and the project already contains a Web site, the Web site cannot be installed if it does not have any virtual directories. In order to be able to install the Web site without virtual directories, you must manually delete it from your InstallShield 2026 Express Edition project, and then re-add it to your project as a new Web site.

This enhancement applies to Express projects.

To learn more, see Creating a Web Site and Adding an Application or a Virtual Directory.

Simplification of QuickPatch Packages

InstallShield now offers the ability to build streamlined QuickPatch packages, which typically have fewer new subfeatures and built-in InstallShield custom actions than those built in earlier releases of InstallShield. A new Streamline QuickPatch setting on the Advanced tab in QuickPatch projects lets you specify whether InstallShield should create this new simpler type of QuickPatch package.

For more information, see:

- Specifying Whether to Streamline the QuickPatch Package
- Advanced Tab

Ability to Password-Protect Patches and QuickPatch Packages

InstallShield now has new password settings that let you password-protect QuickPatch packages. These settings are on the Advanced tab of QuickPatch projects.

If you password-protect a QuickPatch package, any end user who wants to install the package must enter a case-sensitive password to launch your update.

To learn more, see the following:

- Password-Protecting a QuickPatch Package
- Advanced Tab

Ability to Use the /v Command-Line Parameter More than Once to Pass More than One Parameter from Setup.exe to the .msi File

If you want to pass more than one argument from `Setup.exe` to `Msiexec.exe`, you can use the `/v` option multiple times at the command line, once for each argument. Previously, the `/v` option could be used only once, and all parameters were passed through this instance.

This enhancement applies to Express projects.

For more information, see Setup.exe.

Predefined System Searches for the .NET Framework

InstallShield has several new predefined system searches:

- Microsoft .NET Framework 3.5
- Microsoft .NET Framework 3.0 SP1
- Microsoft .NET Framework 3.0
- Microsoft .NET Framework 2.0 SP1
- Microsoft .NET Framework 2.0
- Microsoft .NET Framework 1.1
- Microsoft .NET Framework 1.0

If your installation requires any of these, you can use the Requirements view or the Installation Requirements page in the Project Assistant to add these system searches to your project. When end users launch your installation, Windows Installer checks the target system to see if the requirements are met; if they are not met, the installation displays the error message that is defined for the system search.

This enhancement applies to Express projects.

What's New in InstallShield 2008 Express Edition

New Features

InstallShield 2008 Express Edition includes the following new features.

New End-User Dialog Themes for Express Projects

Dialog themes are predefined sets of images that give your end-user dialogs a unified and distinctive look. By changing the theme option selected in the new Global Dialog Theme setting in the Dialogs view, you can now select one of the available themes for your project, and InstallShield applies that theme to all of the interior and exterior dialogs, as well as the `Setup.exe` initialization dialog, in your project.

For more information about this feature, see the following:

- Dialog Themes
- Selecting or Changing a Dialog Theme
- Global Dialog Settings for All End-User Dialogs

Digital Signing Improvements

InstallShield now lets you digitally sign any files—including your product's executable files—in your installation at build time. In addition, you can now use a personal information exchange file (.pfx) for digital signatures. The Express project type supports this functionality.

The new Signing tab in the Releases view is where you specify the digital signature information—including the digital certificate files that a certification authority grated to you—that InstallShield should use to sign your files. The Signing tab is also where you specify which files in your installation should be digitally signed.

If you specify a .pfx file for signing, InstallShield uses `SignTool.exe` to sign your files. If you specify an .spc file and a .pvk file, InstallShield uses `Signcode.exe` to sign your files. Using a .pfx file is often the preferred method, since it is more likely to work in many different environments (such as locked build machines). If you specify the digital signature password in InstallShield, you will never see a password prompt if you are using a .pfx file. However, if you are using .spc and .pvk files, a password prompt may be displayed.

Previously, InstallShield included support for signing only the .msi and `Setup.exe` files. In addition, InstallShield allowed you to specify .spc and .pvk files for the digital signature, but not .pfx files.

To learn more, see the following:

- Digital Signing and Security
- Digitally Signing a Release and Its Files at Build Time
- Signing Tab

Support for Internet Information Services (IIS) 7.0 and SSL

InstallShield now includes support for IIS 7.

In addition, InstallShield lets you include an SSL certificate for a Web site in your installation. Including an SSL server certificate enables users to authenticate the Web server, check the validity of the Web content, and establish a secure connection.

For more information, see:

- Internet Information Services View
- Specifying the SSL Certificate for a Web Site
- Version-Specific Information for IIS Support in InstallShield

New Microsoft .NET Framework 3.0 Prerequisite Available

InstallShield now includes a new .NET Framework 3.0 setup prerequisite that you can add to Express projects.

For more information, see Adding .NET Framework Redistributables to Projects.

Visual C++ 8.0 SP1 Merge Modules Available

InstallShield now includes Visual C++ 8.0 SP1 merge modules (version 8.0.50727.762).

Support for the UAC Shield Icon on Dialog Buttons

The Install button on the ReadyToInstall dialog and the Remove button on the ReadyToRemove dialog now have the User Account Control (UAC) shield icon when the installation is run on Windows Vista systems and the installation is not yet running with elevated privileges.

Note that InstallShield is run with elevated privileges. Therefore, if you launch your installation from within InstallShield on a Windows Vista system, it has elevated privileges, and the UAC shield icon is not displayed on the ReadyToInstall and ReadyToRemove dialogs.

For a sample screen of a dialog with the UAC shield icon, see Ready to Install Dialog.

Microsoft SQL Server 2005 Express SP1 Setup Prerequisite Available

InstallShield now includes a setup prerequisite for Microsoft SQL Server 2005 Express Edition SP1. You can add this setup prerequisite to Express projects.

Updated DirectX 9.0c Objects

The DirectX 9.0c object now installs all of the latest DirectX 9.0c core and optional components.

In addition, some changes have been made to the DirectX 9 Object Wizard. The wizard now lets you specify whether the redistributable files should be included in the Disk1 folder or streamed into the .msi file. This change enables you to use the DirectX 9 object in compressed installations. Also, you can now use the DirectX 9 object in silent installations.

The custom action that launches the DirectX installation is now sequenced in the Execute sequence and run in deferred system context so that it can be run with elevated privileges on Windows Vista systems.

To learn more, see:

- Including the DirectX 9.0 Object
- DirectX Object Wizard

Ability to Target Windows Server 2008 Systems

InstallShield enables you to specify that your installation requires Windows Server 2008. It also lets you build Windows Server 2008–related conditions for features and custom actions.

New MSXML 6 SP1 Setup Prerequisites Available

InstallShield now includes a new MSXML 6.0 SP1 setup prerequisite that you can add to Express projects.

FlexNet Connect Support

You can add a redistributable for FlexNet Connect 6.1 or 5.x to Express projects. The Update Notifications view lets you select which version of FlexNet Connect you want to include in your project. You can include version 6.1 or any legacy version that is installed in any of the locations that are specified in the Merge Module Location area on the Merge Module tab of the Options dialog box.

The Update Notifications view includes a new Vendor Database setting, which FlexNet Connect 6.1 supports.

Enhancements

InstallShield 2008 Express Edition includes the following enhancements.

Usability Enhancements for Releases

The release settings are now organized by category on several different tabs in the Releases view, which was formerly called the Build Your Release view.

The settings in the Distribute Your Release view have been moved to the new Postbuild tab in the Releases view. The Postbuild tab lets you configure settings for distributing releases to a folder or FTP site automatically at build time.

A new Distribute command is available when you right-click a release in the Releases view. When you select this command, InstallShield copies all of the relevant files for your release to the locations that are specified on the Postbuild tab.

To learn more, see:

- Build Tab
- Setup.exe Tab
- Signing Tab
- .NET/J# Tab
- Internet Tab
- Events Tab
- Testing and Running Installations
- Distributing Releases to a Folder or FTP Site Automatically

Usability Enhancements for the Files view, the Registry View, and the Redistributables View

Several enhancements are available for the Files view:

- You can right-click a file in the **Destination computer's files** pane and then click the new Open Containing Folder command. Doing so opens a Windows Explorer window and displays the folder that contains the file that you right-clicked.
- A new Add command is available when you right-click the **Destination computer's files** pane. Use this command to display an Open dialog box that lets you browse to the file that you want to add to your project.
- The upper-right corner of this view has a new link (either Show Source Panes or Hide Source Panes). Use this new link to show or hide the two top panes—the **Source computer's folders** pane and the **Source computer's files** pane—in this view. You can hide the two panes, open a Windows Explorer window, and drag and drop files from the Windows Explorer window to the two remaining panes in InstallShield.

The Registry view also has a new link (either Show Source Panes or Hide Source Panes) in the upper-right corner. Use this new link to show or hide the two top panes—the **Source computer's folders** pane and the **Source computer's files** pane—in this view.

Two enhancements have also been made to the Redistributables view:

- The right pane in this view shows details about the merge module, object, or setup prerequisite that is selected in the upper-left pane. You can now hide or show the details pane by clicking the Show Details or Hide Details link in the upper-right corner of this view.
- The Details pane that is displayed for setup prerequisites now shows complete information about the selected setup prerequisite. This includes conditions, command-line parameters, and other information that is configured for the prerequisite.

Shortcuts/Folders View Enhancements

Some enhancements have been made to the Shortcuts/Folders view.

- To change the icon that is used for a shortcut, you can right-click the shortcut and then click the new **Change Shortcut icon** command. InstallShield opens the Change Icon dialog box, which enables you to select the icon file and associated icon index that should be used when the shortcut is created on target systems at run time.
- Shortcuts that are listed in the Shortcuts explorer now show the icon image that will be used on the target system. Previously, the Shortcuts explorer used a different image for all types of shortcuts, even if an icon was specified for the shortcut.

For more details about these enhancements, see:

- Specifying the Icon for a Shortcut
- Shortcuts/Folders View

Enhancement for Setup Prerequisites

The Setup.exe tab in the Releases view has a new Setup Prerequisites Location setting that lets you specify where the setup prerequisites in your Express project should be located for the selected release.

The default value is Follow Individual Selections; with this option, InstallShield uses the locations that are specified in the Redistributables view for each individual prerequisite.

The other available options are Download from the Web, Extract from Setup.exe, and Copy from Source Media. These three options override the locations that are specified in the Redistributables view for each setup prerequisite.

For more information, see:

- Specifying a Run-Time Location for a Specific InstallShield Prerequisite
- Specifying the Run-Time Location for InstallShield Prerequisites at the Release Level
- Setup.exe Tab

Enhanced Support for the SecureCustomProperties Property

If you set a public property in the user interface sequence of an installation that requests elevated privileges for the execute sequence, and you want to pass the property's value to the execute sequence, the property must be listed as a value for the `SecureCustomProperties` property, or it must be a restricted public property.

InstallShield now automatically adds to the `SecureCustomProperties` property properties that may need to be passed from the user interface sequence to the execute sequence. To learn more, see Specifying that a Public Property Should Be a Restricted Public Property.

Automatic Downgrade Prevention Entries in Express Projects

To prevent end users from being able to install the current version of your product over a future major version of the same product, InstallShield automatically adds support for preventing the current installation from overwriting a future major version. To learn more, see Preventing the Current Installation from Overwriting a Future Major Version of the Same Product.

Changes for ALLUSERS and for the Customer Information Dialog

Beginning with InstallShield 2008 Express Edition, the ALLUSERS property is set to 1 by default in all new Express projects. This is the recommended implementation, since most installations must be run in a per-machine context with administrative privileges.

If you upgrade a project that was created with InstallShield 12 Express Edition or earlier toInstallShield 2026 Express Edition, InstallShield does not automatically change the value of the ALLUSERS property or add this property if it was not defined in the earlier project. The General Information view has a new ALLUSERS setting that lets you set the value of ALLUSERS.

Also new with InstallShield 2008 Express Edition, by default, the Customer Information dialog in all new Express projects does not display the radio button group that enables end users to specify whether they want to install the product for all users or for only the current user. This is the recommended implementation for this dialog.

If you upgrade a project that was created with InstallShield 12 Express Edition or earlier to InstallShield 2026 Express Edition, InstallShield does not automatically change the Customer Information dialog. You can use the Dialogs view to show or hide the radio group button for this dialog.

To learn more, see:

- Per-User vs. Per-Machine Installations
- General Information Settings
- ALLUSERS
- Customer Information Dialog

Ability to Change the Product Version from the Command Line or Through an MSBuild Task Parameter

The -y command-line parameter is available for command-line builds with IsCmdBld.exe. Use this parameter to specify a product version from the command-line build.

In addition, the InstallShield task for MSBuild now includes a ProductVersion parameter, which you can use to specify the product version through MSBuild. This parameter is exposed as the property InstallShieldProductVersion when the default targets file is used.

Using the -y command-line parameter or the InstallShield task ProductVersion parameter is especially helpful if you want to increment the build version (the third field) of the product version.

To learn more, see:

- IsCmdBld.exe
- Microsoft Build Engine (MSBuild)

New Setting for Specifying Whether an IIS Web Server Should Allow the CMD Command to Be Used for SSI #exec Directives

You can configure an IIS Web server to prevent the CMD command for the #exec directive from being used to execute shell commands, or you can configure it to allow the CMD command to be used to execute this type of command. The SSIEnableCmdDirective registry value for the HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W3SVC\Parameters registry key is what determines whether the CMD command is permitted.

The Internet Information Services view in InstallShield includes a new SSIEnableCmdDirective registry value setting. This setting lets you specify how your installation should configure the SSIEnableCmdDirective registry value on target systems. This setting also lets you specify that the SSIEnableCmdDirective registry value should not be changed at installation run time; this is the default behavior.

For more information, see Specifying Whether a Web Server Should Allow the CMD Command to Be Used for SSI #exec Directives.

New Host Header Name Setting for IIS Web Sites

You can now use the new Host Header Name setting on the Web Site tab for a Web site in the Internet Information Services view to specify the host header name that identifies the IIS Web site that is installed during your installation.

For more information, see Specifying the IIS Host Header Name for a Web Site.

Ability to Remove Unreferenced Directories from the .msi File

The Build tab in the Releases view includes a new Keep Unused Directories setting. Use this setting to specify whether you want InstallShield to remove unused directories from the `Directory` table of the .msi file when you build the selected release. The default value is No.

This setting is available for Express projects.

For more information, see Build Tab.

New Check Box for Specifying Whether COM+ Applications Should Be Installed After the InstallFinalize Action

The Installation tab in the Component Services view has a new **Install after InstallFinalize action** check box. If a selected COM+ application in your project contains .NET assemblies that need to be installed to the global assembly cache (GAC), select this check box. If you select this check box, the ISComponentServiceFinalize action installs the selected COM+ application after the InstallFinalize action. Windows Installer does not commit changes made in the in-script session to the GAC until InstallFinalize.

Additional Predefined System Searches for Express Projects

InstallShield has several new predefined system searches:

- Adobe Reader 7
- Adobe Reader 6
- Internet Explorer 7.0

If your installation requires any of these products, you can use the Requirements view or the Installation Requirements page in the Project Assistant to add these system searches to your project. When end users launch your installation, Windows Installer checks the target system to see if the requirements are met; if they are not met, the installation displays the error message that is defined for the system search.

Enhancements for Patch Display Information

The Identification tab, which was previously called the Uninstall tab, is where you specify information that should be displayed for a QuickPatch package in Add or Remove Programs on systems running Windows Installer 3.0 or later. This tab in the General Information view in QuickPatch projects has settings for items such as the display name, the manufacturer name, and the support URL. Now every time that you change the latest setup for a QuickPatch project, InstallShield uses the Add or Remove Programs information from the latest setup as the values for the Identification tab settings. You can override the values on the Identification tab as needed. In addition, the **Allow Patch to Be Uninstalled (Requires Windows Installer 3.0)** check box is now available on the Common tab. This setting was previously available on the Uninstall tab.

For more information, see:

- Common Tab (in a QuickPatch project)
- Identification Tab (in a QuickPatch project)

Ability to Specify the Minimum Initialization Time

InstallShield includes a new Minimum Initialization Time setting on the Setup.exe tab for a release in the Releases view. Use this setting to specify the minimum number of seconds that the installation should display the initialization dialog—as well as the splash screen, if one is included—when end users run this release.

For more information, see Setup.exe Tab.

What's New in InstallShield 12 Express Edition

InstallShield includes the following new features:

Ability to Target Windows Vista Systems

InstallShield enables you to specify that your installation requires Windows Vista. It also lets you build Windows Vista–related conditions for features.

User Account Control Support

InstallShield includes support for the User Account Control functionality that Microsoft added for Windows Vista. Use the new Require Administrative Privileges setting in the General Information view to specify at a project-wide basis whether administrative privileges are required for an installation. Also, use the Required Execution Level setting in the Build Your Release view to specify the minimum level required by your installation's `Setup.exe` file for running the installation (the setup launcher, any setup prerequisites, and the .msi file) on Windows Vista platforms.

For details, see:

- General Information Settings
- Specifying the Required Execution Level for Your Setup Launcher on Windows Vista and Later Platforms

Digital Signature Enhancements

If you specify digital signature information for your installation, InstallShield automatically adds the necessary information to the `MsiDigitalCertificate` and `MsiPatchCertificate` tables. The **MsiPatchCertificate** table contains the information that is needed to enable User Account Control (UAC) patching. This enables you to create QuickPatch packages that can be applied by non-administrators.

In addition, the Build Installation page in the Project Assistant now offers the ability to specify digital signature information for your installation. Also, InstallShield now enables you to specify digital signature information for all media types in the Build Your Release view. Previously, only the WebDeployment media type could include digital signature information.

For more information, see:

- Preparing Installations for Non-Administrator Patches
- Digital Signing and Security

Support for Minimizing Reboots Through the Restart Manager Infrastructure

Restarting the system after an installation is inconvenient for end users. One of the Certified for Windows Vista program requirements is that all installations must contain an option that enables end users to automatically close applications and attempt to restart them after the installation is complete.

To support this quality guideline, an MsiRMFilesInUse dialog is available in all Express projects. The installation displays this dialog if one or more files that needs to be updated are currently in use during the installation. To learn more, see:

- Minimizing Reboots on Windows Vista and Later Systems
- MsiRMFilesInUse Dialog

Windows Installer 4.0 Log File Support on a Project-Wide Basis

InstallShield enables you to specify on a project-wide basis—without having to use the command line or configure log parameters through the registry—whether Windows Installer 4.0 should log your installation. You can also customize the types of messages that are logged.

To enable logging, use the new Create MSI Logs setting in the General Information view. You can click the ellipsis button (...) in this setting to display the Logging Options for Windows Installer 4.0 and Later dialog box, which is where you specify whether logging should occur. This is also where you override the default logging parameters if you want to customize the types of messages that are logged.

If you do enable logging, Windows Installer 4.0 creates a log file during installation of your product and populates the `MsiLogFileLocation` property with the log file's path. In addition, a **Show the Windows Installer log** check box is added to the Setup Complete Success, Setup Complete Error, and Setup Interrupted dialogs. If the end user selects that check box and then clicks Finish, the log file is opened in a text file viewer or editor.

For more information, see Specifying Whether Windows Installer Installations Should Be Logged.

Multilingual User Interface (MUI) Support

If you are preparing an installation for a multilingual application and Windows Installer 4.0 will be running the installation, you can now use InstallShield to create shortcuts that include support for the Windows multilingual user interface (MUI). Four new settings are available in the Shortcuts/Folders view for a selected shortcut:

- Display Resource DLL
- Display Resource ID
- Description Resource DLL
- Description Resource ID

These new settings correspond with the four new columns in the Shortcut table for Windows Installer 4.0. To learn more, see Shortcut Settings.

Registry and File Filtering Enhancements for COM Extraction and Dependency Scanners

To prevent InstallShield from extracting undesired COM data from a COM server, you can edit a new `Filters.xml` file that is installed with InstallShield. Editing this `Filters.xml` file enables you to customize the list of registry keys that will be excluded from COM extraction.

The `Filters.xml` file also now lists files that the Static, Dynamic, and Visual Basic dependency scanners will exclude or include. Previously, two different files—`Userscan.ini` and `Iswiscan.ini`—were used to list excluded and included files.

For more information, see:

- Filtering Registry Changes for COM Extraction
- Filtering Files in Dependency Scanners

Enhanced Start Page

The list of recently opened projects that are displayed on the Start Page now includes a column that shows the project type. In addition, the maximum number of projects that are listed has been increased from four to eight.

Workaround for Windows Installer Issue with Upgrades and Assemblies

By default, upgrades that are created in the Upgrade Paths view are configured to remove the earlier version of the product before installing the new version. That is, the RemoveExistingProducts action is scheduled before the InstallFinalize action. This default sequencing behavior may cause a problem if the product includes an assembly that is installed to the global assembly cache (GAC): after the upgrade is applied, the assembly may be missing from the GAC. The issue occurs because Windows Installer cannot properly reference count the assembly; therefore, it is removed but not reinstalled during the upgrade.

To work around this Windows Installer issue, you can now configure your project in InstallShield so that the new version of your product is installed before the earlier version is removed.

To learn more, see Preventing the Removal of Assemblies from the Global Assembly Cache During Upgrades.

Target System Requirements

You can use InstallShield to rapidly build, test, and deploy installations that target Windows-based systems.

Requirements For Desktop Computers

Operating System

Target systems must meet the following minimum operating system requirements:

- Windows 7
- Windows Server 2008 R2
- Windows 8
- Windows Server 2012
- Windows Server 8.1
- Windows Server 2012 R2
- Windows 10
- Windows Server 2016
- Windows Server 2019
- Windows 11
- Windows Server 2022
- Windows Server 2025

Target systems must also support the SSE2 instruction set.

Installer Engine Requirements

The minimum target system requirements for the various installer engines are as follows.

Table 1-2 ▪ Target System Requirements for Desktop Computers

Installer Engine	Operating System and Other Requirements
Windows Installer 5.0 **(This is not available as a redistributable.)**	Windows 7 or later, Windows Server 2008 R2 or later
Windows Installer 4.5	Windows XP SP2 or later, Windows Server 2003 SP1 or later
Windows Installer 4.0 **(This is not available as a redistributable.)**	Windows Vista, Windows Server 2008
Windows Installer 3.1	Windows 2000 SP3 or later, Windows Server 2003 or later
Windows Installer 3.0	Windows 2000 SP3 or later, Windows Server 2003 or later

Table 1-2 ▪ Target System Requirements for Desktop Computers

Installer Engine	Operating System and Other Requirements
Windows Installer 2.0	Windows 95 or later (Note that InstallShield does not include support for Windows 95, Windows 98, Windows NT 4, or Windows Me.)

Challenges of Supporting Both 32-Bit and 64-Bit Target Operating Systems

If you want your product to support both 32-bit and 64-bit target systems, note the following fundamental Windows Installer conflict when you are creating your product's installation:

- A 32-bit Windows Installer package cannot install to 64-bit file or registry locations.
- A 64-bit Windows Installer package cannot run on 32-bit systems.

Windows Installer does not have support for creating a multi-architecture Windows Installer package that installs to 32-bit locations on 32-bit systems and 64-bit locations on 64-bit systems.

Key Differences Between 32-Bit and 64-Bit Support

32-Bit vs. 64-Bit Windows Installer Packages

A Windows Installer package is considered to be 32 bit if its Template Summary property indicates Intel; it is considered to be 64 bit if its Template Summary property indicates x64. A 32-bit Windows Installer package can run on 32-bit systems and on most 64-bit systems. However, a 64-bit Windows Installer package runs only on 64-bit systems. Running a 64-bit Windows Installer package on a 32-bit system results in run-time error 1633:

```
This installation package is not supported by this processor type. Contact your product vendor.
```

The Express edition of InstallShield configures the Template Summary property automatically.

32-Bit vs. 64-Bit Components

A component is the smallest installable part of a product. It contains files, registry entries, and other data to be installed on target systems. (For more information, see Installation Fundamentals.) The Express edition of InstallShield creates each component for a Windows Installer package automatically at build time, and marks each component as 64 bit if appropriate.

A 32-bit Windows Installer package cannot contain any 64-bit components, but a 64-bit Windows Installer package can contain 32-bit components, 64-bit components, or a mix of both.

In order for Windows Installer to install to a 64-bit location on a 64-bit system, the component in the 64-bit Windows Installer package that contains the data for the 64-bit location must be marked as 64 bit. If a component in a 64-bit Windows Installer package is not marked as 64 bit, its data is installed to 32-bit locations.

A 32-bit component in a 32-bit Windows Installer package installs to 32-bit locations on 32-bit and 64-bit systems. A 32-bit component in a 64-bit Windows Installer package installs to 32-bit locations on 64-bit systems.

32-Bit vs. 64-Bit File Locations

Microsoft designed 64-bit versions of Windows to allow existing 32-bit applications to continue to work seamlessly. They also designed 64-bit versions of Windows in such a way to allow a recompiled version of the same code to work seamlessly as a 64-bit application. To provide this support, 64-bit versions of Windows isolate the 32-bit and 64-bit files from each other by storing their files in separate locations.

A 64-bit target system typically has two Program Files folders:

- Program Files, which is for 64-bit applications
- Program Files (x86), which is for 32-bit applications

A 64-bit target system typically has two Common Files folders (one in either Program Files folder):

- Program Files\Common Files, which is for 64-bit applications
- Program Files (x86)\Common Files, which is for 32-bit applications

A 64-bit target system also typically has two system folders:

- System32, which is for 64-bit libraries and executable files
- SysWOW64, which is for 32-bit libraries and executable files

If a 32-bit Windows Installer package is configured to install a product's files and folders to the Program Files folder, Windows Installer installs those files and folders in the Program Files folder on 32-bit systems. However, if end users run the 32-bit installation on 64-bit systems, Windows Installer installs the files and folders to the Program Files (x86) folder.

Similarly, if a 32-bit Windows Installer package is configured to install any of a product's files and folders to the System32 folder, Windows Installer installs those files and folders to System32 on 32-bit systems and to SysWOW64 on 64-bit systems.

32-Bit vs. 64-Bit Registry Locations

The isolation of 32-bit and 64-bit data from each other on 64-bit systems also occurs in the registry. A 64-bit target system typically has two HKEY_LOCAL_MACHINE\Software keys:

- HKLM\Software, which is for 64-bit applications
- HKLM\Software\Wow6432Node, which is for 32-bit applications

If a 32-bit Windows Installer package is configured to install a product's registry data under the HKLM\Software key, Windows Installer installs that data under the HKLM\Software key on 32-bit systems, but under HKLM\Software\Wow6432Node on 64-bit systems.

Tip ▪ *To see how a 32-bit application views the registry on a 64-bit system, launch the 32-bit version of the Registry Editor (the* `regedit.exe` *file in the SysWOW64 folder).*

Deciding Between 32-Bit and 64-Bit Windows Installer Packages

Most installation authors choose to create 32-bit Windows Installer packages for all of their end users. Typically the 32-bit packages can be successfully run on their end users' machines and their products run as designed, regardless of whether those end users have 32-bit or 64-bit versions of Windows.

If your product consists of 32-bit product files for 32-bit systems and equivalent 64-bit product files for 64-bit systems, you may want to create a 32-bit Windows Installer package for end users who install on 32-bit systems and a separate 64-bit Windows Installer package for end users who install on 64-bit systems.

If your product targets only end users who have 64-bit versions of Windows, you may prefer to create a 64-bit Windows Installer package instead of a 32-bit one.

32-Bit and 64-Bit Support in InstallShield

InstallShield includes support for creating the following types of installations:

- A 32-bit Windows Installer package that installs to 32-bit file and registry locations.
- A 64-bit Windows Installer package that installs to 64-bit file and registry locations. This type of package can also install to 32-bit file and registry locations.

How InstallShield Determines Whether to Build a 64-Bit or 32-Bit Windows Installer Package

If one or both of the following conditions are true at build time, InstallShield creates a 64-bit Windows Installer package (in which x64 is specified for the Template Summary property):

- The Files view in the Express project contains one or more folders or files that are configured to be installed to `CommonFiles64Folder`, `ProgramFiles64Folder`, or `System64Folder`.
- The Registry view in the Express project contains one or more registry entries under the HKEY_LOCAL_MACHINE\SOFTWARE (64-Bit) node.

In addition, when InstallShield builds a 64-bit Windows Installer package, it marks the components that contain the files, folders, and registry entries in 64-bit locations as 64 bit.

If neither of the aforementioned conditions is true, InstallShield creates a 32-bit Windows Installer package (in which Intel is specified for the Template Summary property). In addition, none of the components in the package are marked as 64 bit. In this type of scenario, all of the files, folders, and registry entries are installed to 32-bit locations, even on 64-bit target systems.

***Edition** ▪ Note that some 64-bit target systems—such as Windows Server Core systems—may not have 32-bit Windows-on-Windows (WOW64) support. These 64-bit target systems cannot run 32-bit Windows Installer packages. The Premier and Professional editions of InstallShield include support for creating pure 64-bit Windows Installer packages that work on these systems.*

Also note that the Premier and Professional editions of InstallShield include support for creating two Windows Installer packages (one 32 bit and one 64 bit) from a single project file, and for using release flags to include or exclude various parts of the project in each release at build time. If you want to create a 32-bit Windows Installer package and a 64-bit Windows Installer package in the Express edition of InstallShield, you must create two separate Express projects.

The Premier edition of InstallShield includes support for a Suite/Advanced UI installation that lets you simplify the distribution of a product's installation, since it enables you to combine 64-bit and 32-bit Windows Installer packages into a single installation; the appropriate package is run on each target system at run time.

Including 64-Bit Merge Modules in Express Projects

InstallShield lets you add 64-bit merge modules to Express projects. If you add a 64-bit merge module that has one or more 64-bit components to an Express project, the Express project must have at least one file, folder, or registry entry that is configured to be installed to a 64-bit location; otherwise, the following error occurs at build time:

```
error -5008: This 32-bit package cannot include 64-bit data. The 64-bit data may come from a merge
    module.
```

If the Express project does not have at least one file, folder, or registry entry that is configured to be installed to a 64-bit location, InstallShield attempts to create a 32-bit Windows Installer package at build time; however, a 32-bit Windows Installer package cannot contain a 64-bit merge module that contains one or more 64-bit components.

Building 64-Bit Setup Launcher

Launching InstallShield with vs. Without Administrative Privileges

If you launch InstallShield without administrative privileges, the following functionality is not available:

- **COM extraction**—Extracting COM information from a COM server requires administrative privileges.

  If you specify that you want to have COM information extracted from a COM server in your project, and then you try to build a release while running InstallShield without administrative privileges, build error -6017 occurs.

- **Redistributable downloading**—Downloading redistributables from within the Redistributables view requires administrative privileges. This is because InstallShield tries to download the files to a per-machine location, which requires administrative privileges.

  If you try to download a redistributable from within the Redistributables view but you do not have administrative privileges, InstallShield displays the following message:

  ```
  The download failed; make sure you are running as Administrator, and that your machine is connected
      to the Internet. Would you like to try again?
  ```

- **Ability to specify All Users locations for InstallShield prerequisites**—The Prerequisites tab on the Options dialog box is where you specify the folders that contain the InstallShield prerequisites that should be displayed in the Redistributables view. Modifying the All Users location on that tab requires administrative privileges, since InstallShield writes the information to a per-machine location in the registry. Therefore, the All Users location on that tab is disabled if you are running InstallShield without administrative privileges.

- **Ability to specify All Users locations for merge modules**—The Merge Module Options tab on the Options dialog box is where you specify the folders that contain the merge modules that should be displayed in the Redistributables view. Modifying the All Users location on that tab requires administrative privileges, since InstallShield writes the information to a per-machine location in the registry. Therefore, the All Users location on that tab is disabled if you are running InstallShield without administrative privileges.

- **Ability to edit the locations for Regasm.exe and InstallUtilLib.dll**—The .NET tab on the Options dialog box is where you specify the locations of `Regasm.exe` and `InstallUtilLib.dll` files, which are utilities that are included with the .NET Framework. These utilities are used for COM interop and .NET custom actions.

Modifying these locations on the .NET tab requires administrative privileges, since InstallShield writes the information to a per-machine location in the registry. Therefore, these location settings on that tab are disabled if you are running InstallShield without administrative privileges.

If you switch between full Administrator and non-Administrator contexts and you use mapped-drive locations in your projects, you may encounter issues. For example, if you map a drive letter to a shared network folder through Windows Explorer without administrative privileges, you can access this drive letter in a non-administrative instance of InstallShield, but not in an administrative instance. Likewise, if you map a drive letter to a shared network folder through Windows Explorer with administrative privileges, you can access this location in an administrative instance of InstallShield, but not in a non-administrative instance. Thus, if you want to reference network locations in your project, consider using UNC paths (such as \\server\share) or mapping drive letters both with and without administrative privileges.

Note that if you are using InstallShield from within Visual Studio, you may not have administrative privileges. By default, if you launch Visual Studio by double-clicking its shortcut on a Windows Vista or later system, you will not have administrative privileges.

Task ***To run InstallShield from within Visual Studio with administrative privileges on a Windows Vista or later system:***

1. On the Start Menu, right-click the Visual Studio shortcut and then click **Run as administrator**.
2. Create a new InstallShield project or open an existing one. For more information, see one of the following topics:
   - Creating InstallShield Projects in Microsoft Visual Studio
   - Opening InstallShield Projects in Microsoft Visual Studio

Developing and Building Installations on 32-Bit vs. 64-Bit Systems

InstallShield is a 32-bit application that runs on both 32-bit and 64-bit systems. In some cases, InstallShield behaves differently, depending on whether you are using it on a 32-bit system or a 64-bit system; you may also encounter differences depending on which operating system you are using. The following sections explain these differences.

Note that like InstallShield, the command-line build (`ISCmdBld.exe`)is also a 32-bit application. Therefore, the same platform-specific differences occur for this tool.

Adding System Files to Your Project

When you are using InstallShield on a 64-bit system and you want to add to your project a system file whose source location is the 64-bit System folder (System32) on your development machine, it is not possible to drag the file from one of the source computer panes at the top of the Files view to the appropriate location in one of the destination computer panes.

To add a 64-bit System32 file on your development machine to your InstallShield project, you can browse to the Sysnative folder on your machine and then select the appropriate file for your project. To learn more, see Adding Files from Your 64-Bit Source Machine's 64-Bit System32 Folder.

Viewing Both the 32- and 64-Bit Areas of the Source Machine's Registry on 64-Bit Development Systems

If you are using InstallShield on a 64-bit development system, the Registry view in InstallShield displays both the 32-bit and 64-bit areas of your machine's registry:

- HKEY_LOCAL_MACHINE\Software
- HKEY_LOCAL_MACHINE\Software\Wow6432Node

This support enables you to drag and drop entries from those source areas to the appropriate areas in the destination pane of this view when you are configuring registry data changes for your project.

Note that if you want your installation to install registry data to a 64-bit area of the registry on 64-bit target systems without having it redirected to a 32-bit area, you must place the registry data in the HKEY_LOCAL_MACHINE\SOFTWARE (64-Bit) node in the destination pane in the Registry view. Simply dragging 64-bit data from the source panes in the Registry view to a non-64-bit location in one of the destination panes of the view does not mark the component as 64 bit. For more information, see Challenges of Supporting Both 32-Bit and 64-Bit Target Operating Systems.

Scanning 64-Bit Files for Dependencies

If you use the dependency scanners in InstallShield to help you identify dependencies that you may need to add to your project, these scanners can check 64-bit files in your project for dependencies only if you are using InstallShield on a 64-bit operating system.

If you are using InstallShield on a 32-bit system, the dependency scanners in InstallShield cannot check 64-bit files in your project for dependencies. To add 64-bit dependencies to a project on a 32-bit system, consider manually adding the required files and merge modules to your project.

To learn about scanning files for dependencies, see Identifying Application Dependencies.

Using 64-Bit .NET Installer Classes and COM Interop

If you are using InstallShield on a 64-bit version of Windows, the .NET tab on the Options dialog box—which is displayed when you click Options on the Tools menu in InstallShield—lets you specify different paths for the 32-bit and 64-bit locations of the `Regasm.exe` and `InstallUtilLib.dll` files that are included with the .NET Framework. InstallShield uses the paths that you specify at build time for releases that include .NET installer classes and COM interop.

If you are building from the command line with `ISCmdBld.exe` on a 64-bit version of Windows, and you use the existing `-t` parameter to specify the path of the 32-bit version of the .NET Framework, `ISCmdBld.exe` uses the 64-bit location of `Regasm.exe` and `InstallUtilLib.dll` for 64-bit .NET installer classes and COM interop.

If you are building through MSBuild or Team Foundation Server (TFS) and you use the existing DotNetUtilPath parameter on the InstallShield task to specify the path of the 32-bit version of the .NET Framework, the build uses the 64-bit location of `Regasm.exe` and `InstallUtilLib.dll` for 64-bit .NET installer cases and COM interop.

Note that if you are using InstallShield on a 32-bit operating system, the setting for the 64-bit location is disabled, and only 32-bit support for .NET installer cases and COM interop is available. This support also applies to building from the command line with `ISCmdBld.exe`, and building through MSBuild or TFS.

To learn more, see:

- .NET Tab (on the Options dialog box)

- IsCmdBld.exe
- Microsoft Build Engine (MSBuild)

Using Help

Revenera understands the importance of having useful information and help resources at your fingertips. In addition to the inline help embedded within various views of the InstallShield interface, InstallShield includes a help library that is installed with the product, an online help system that is available on the Web, and documentation that is available in PDF format.

InstallShield Help Library

When you have questions about your product, first consult the InstallShield Help Library. It is a complete user guide.

You can access the InstallShield Help Library from the **Help** menu in InstallShield, by pressing F1, or by clicking Help buttons in the interface.

No Internet connectivity is required to view the InstallShield Help Library. Essentially, the online help viewer is a tool that you can use to display, search, and filter technical information based on your personal needs.

Using Context-Sensitive Help

While working on a project, clicking a software object displays its help information in the Help window. This is also called “context-sensitive” help.

Help information is also available for each of the properties of a selected software object displayed in the Explorer window, which provides instructions for setting that property.

Web-Based Online Help

Web-based online help is available to you 24 hours a day, seven days a week, on our Web site at http://docs.revenera.com. This help resource center provides near real time updates to documentation.

Documentation as PDF

InstallShield documentation is also available as a PDF at: https://community.revenera.com/s/.

Contacting Us

Revenera is headquartered in Itasca, Illinois, and has offices worldwide. To contact us or to learn more about our products, visit our Web site at:

http://www.revenera.com

2

Getting Started

InstallShield provides powerful features and time-saving tools that make authoring reliable Windows Installer installations easy. The InstallShield Help Library is a resource that will help you harness the full potential of InstallShield. The help topics you might want to read first depend upon your previous experience with InstallShield installation-authoring software. The table below directs you to various topics based on your level of experience.

Table 2-1 ▪ Getting Started Road Map

Level of Familiarity	Top Help Topics
You have not created installations previously.	To learn general information about installations, refer to the following topics: ● Installation Fundamentals ● Application Lifecycle
You are new to InstallShield.	If you have created installations previously but you do not have experience using InstallShield, see the following topics: ● Working with the InstallShield Interface ● Working with Projects ● Tutorials ● Supported Application Programming Languages

Table 2-1 ▪ Getting Started Road Map (cont.)

Level of Familiarity	Top Help Topics
You have used other Revenera products.	If you have used other products from Revenera, refer to any of the following topics: ● Upgrading from Earlier Versions of InstallShield ● Upgrading to the Premier or Professional Editions of InstallShield ● Project Assistant ● Overview of Installations
You are an intermediate-level or advanced-level user of InstallShield.	If you are familiar with InstallShield, you may want to review the following topics: ● Errors and Warnings

Installation Fundamentals

An installation, in its simplest terms, is the "package" used to install your files and programs onto your user's machine. It is a complete collection of the application files, as well as logic that interacts with the installer engine. The primary task of any installation is to transfer the application files from the source medium to the end user's computer. The complexities of the Windows operating system make it anything but simple to create an effective, coherent installation without the aid of a utility such as InstallShield.

An installation is divided into three levels: products, features, and components. The following diagram illustrates this hierarchy:

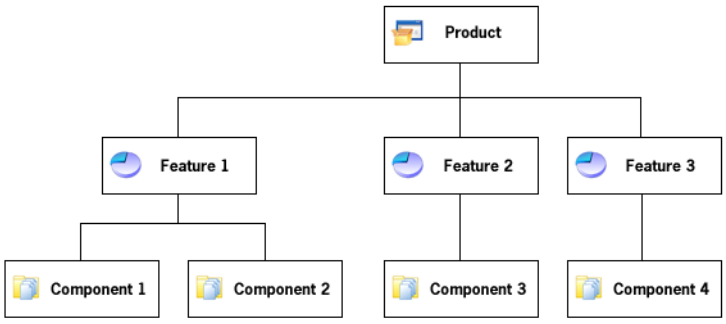


Figure 2-1: Installation Hierarchy

A *product* is the highest level of organization in an installation project. A product is usually one main application (for example, a word processor) and all of the files and data that the application requires; a suite of applications may also be a product.

A *feature* is the smallest installable part of a product, from the end user's perspective. As the designer of an installation program, you usually allow the user to choose which features to install and which features to leave on the source media. In a word processor product, the main executable may be one feature, and optional dictionaries may be a separate feature. A feature should be self-contained, in the sense that a feature should not

require sibling features. For example, a thesaurus feature should not require a dictionary feature that the user can choose not to install. However, you can design features to contain subfeatures, which allow the end user finer control over which files and data to install.

Each feature in a project is made up of one or more *components*. A component is the smallest installable part of a product. The Express edition of InstallShield creates components for you automatically. The actual breakdown of features into various components is not visible anywhere within the InstallShield user interface. In addition, components are invisible to the end user. Each component contains files (and other resources) with similar properties. For example, all of the files in a component will be installed in the same directory on an end user's machine, and all of the files in a component should apply to the same operating system or language. A dictionary feature might contain several language-specific dictionary components. In addition to containing files, components generally contain registry data, shortcuts, file extension information, and other system data to write to a user's machine.

Overview of Installations

Even if you are unaware of what a setup is, chances are you have used one before. If you have ever installed a product onto your computer, you have seen a setup in action, from the end user's perspective. The setup transfers files from the source medium to your local drive. It also makes the required registry entries, creates shortcuts, and registers COM servers. Setups commonly gather information about the target machine and the user.

Typical Elements of an Installation

Perform File Transfer

File transfer involves copying files from the source medium, such as a CD or a floppy disk, to a local drive on the end user's machine. Depending on the configuration the end user chooses, all or only some of the files may be transferred to the local disk. During file transfer, a setup can display billboards that provide product information such as new features or usability tips. A status bar may also be displayed to show the progress of the file transfer process.

Display User Interface

The user interface of a setup provides information and setup configuration choices to the end user. Through the user interface an end user can choose to install only part of a product, choose to leave some files on the source medium, view a license agreement, or provide information to the installer that may be necessary to ensure the proper configuration of the installation. The user interface can be customized to meet whatever needs you may have. For example, you can prompt a user for a serial number before starting the installation if you want to protect your software against illegal use.

Create Shortcuts

Shortcuts are links to files and applications that can be created on the end user's machine during a setup. Shortcuts are often placed on the desktop or the Start menu of the target machine to provide quick and easy access to a program or its files.

Register File Associations

If your product uses a distinct file type you need to register those file types on the end user's system. For example, Notepad creates a file with a .txt extension. In order for a file type to be recognized by your end user's system, it must be registered in the system registry. The process of registering a file type is handled during a setup.

Register COM, COM+, and DCOM Files

COM servers (such as ActiveX, COM, and COM+ files) require special registration so that applications can access the files' interfaces. Traditionally, these EXEs, DLLs, and OCXs contained self-registration functions that could be invoked to register the files during installation. However, relying on self-registration can cause some problems: the end user cannot always be certain of what information is being registered or that the registry entries will be fully cleaned up when the file is uninstalled.

The solution provided by the Windows Installer service is to write the necessary registry entries during setup and then remove them when the COM component is uninstalled. This method helps to ensure that all the COM servers are registered in the appropriate way.

Register Product for Uninstallation

In order to uninstall a product, the operating system must know that the product is present. Therefore, a setup registers a product with the operating system so that it can be easily uninstalled. This registration is required for Windows logo compliance. Much of the information registered in this process is available to the end user through Add or Remove Programs in the Control Panel. For example, technical support contact information, product update information, product version, and product publisher information are all registered in this process.

Application Lifecycle

Your application lifecycle should not end when your customer installs your application. As a software vendor, the success of your application goes well beyond the initial installation on the customer's desktop. Customers expect to have easy access to product updates, enhancements, and critical information. Your ability to communicate with your customers and monitor the health of your application is vital to your ongoing growth and profitability.

Too often, software vendors require their customers to initiate communication. Vendors who are not proactively creating an ongoing dialog are missing a tremendous opportunity. Unless the customer is an active visitor to your Web site or public user communities, they miss important information about updates, upgrades, hot fixes, and general technical bulletins. You miss revenue and service opportunities.

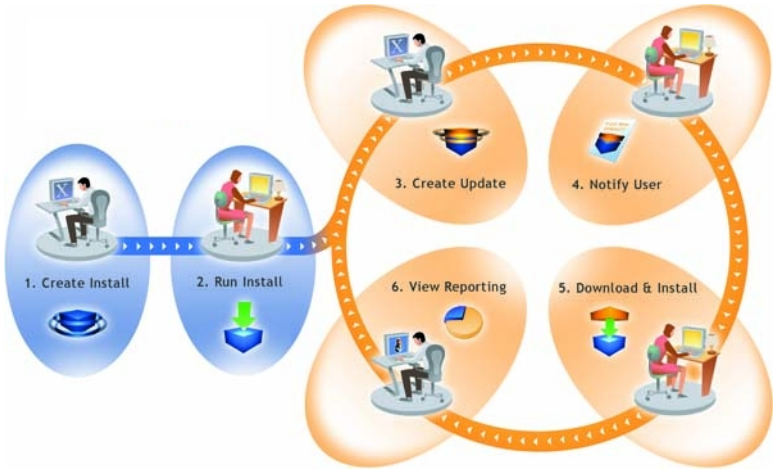


Figure 2-2: How FlexNet Connect Manages the Application Lifecycle

The above diagram illustrates how FlexNet Connect is used to manage the application lifecycle:

1. **Create Install—**InstallShield makes it easy for software developers to create installations that run on any platform.
2. **Run Install—**Installations created with InstallShield technology have successfully installed on over 400 million machines worldwide.
3. **Create Update—**InstallShield enables software developers to rapidly build patches and updates.
4. **Notify User—**FlexNet Connect notifies every user that a new update is ready to be installed.
5. **Download & Install—**FlexNet Connect downloads the update and installs it in one seamless, integrated process.
6. **View Reporting—**FlexNet Connect provides immediate feedback on the update's adoption rate.

Starting InstallShield

Each time you open InstallShield, you are taken directly to the InstallShield Start Page. The Start Page provides quick access to product information, to recently opened projects, and to InstallShield resources.

InstallShield Start Page

The InstallShield Start Page provides quick access to product information, to recently opened projects, and to InstallShield resources. The Start Page includes the following sections:

Table 2-2 ▪ Sections on the Start Page

Section	Description
Project Tasks	Click a project task to quickly create a new project, open an existing project, or browse to one of the sample projects included with the InstallShield installation.

Table 2-2 ▪ Sections on the Start Page (cont.)

Section	Description
Help Topics	Frequently accessed help topics are listed in this section. To access the entire InstallShield Help Library from the Start Page, press F1 or click the Help Library link in the Resources section.
(Recently Opened Projects)	The section in the middle of the Start Page lists your most recently accessed projects, the project types, and the dates on which they were last modified.
Getting Started	Click the Getting Started heading for guidance on what areas of the InstallShield Help Library to read, based on your level of experience with InstallShield and installation-authoring tools.
Resources	The Resources section contains a number of links to connect you to helpful InstallShield information. • **Help Library**—Displays the InstallShield documentation. • **InstallShield Community**—Provides a Web-based forum where you can join other InstallShield users, post questions, and search for answers. • **Webinars**—Directs you to free Web-based seminars that help you evaluate InstallShield and gain the most from your Revenera products. • **Downloads**—Offers a place where you can download the latest InstallShield prerequisites, InstallShield merge modules, and objects; service packs; patches; and more for the version of InstallShield that you are using. • **Release Notes**—Connects you to the release notes that are posted to the Knowledge Base on the Revenera Web site. • **Known Issues**—Displays the Knowledge Base article that lists the known issues for the version of InstallShield that you are using and provides information about workarounds and resolutions. • **Upgrade Alerts**—Displays the Knowledge Base article that provides information about possible issues that may occur when you upgrade projects that were created with earlier versions of InstallShield to InstallShield 2026 Express Edition. It also alerts you to possible changes in behavior that you may notice between new InstallShield 2026 Express Edition projects and projects that are upgraded from earlier versions. • **RSS Feeds**—Directs you to the Web page where you can subscribe to RSS feeds of InstallShield Knowledge Base articles.
Contact Us	To access the Support area of the Revenera Web site, click one of the links listed in this section.

Working with Projects

When you create your installation, you also create an InstallShield project file (.ise). This file stores all the logic and information necessary to build your installation file (.msi). An InstallShield project specifies the files, folders, and operations that make up the project's output. A project's output is the installation.

A project can be as simple or as complex as you need to meet your requirements. A simple project might consist of files, features, and registry entries. More complex projects might consist of these items plus redistributables, initialization file changes, and calls to external .dll file functions.

Project Types

InstallShield offers a number of different project types to assist you in creating the optimal project for your end users.

- **Express Project**—An Express project uses Windows Installer to provide the user interface for the installation. When you choose this project type, you need to create features and specify all application files and other distributable data.
- **QuickPatch Project**—This project type is recommended for installation authors who want to ship small, single updates to their end users. Creation of a QuickPatch begins with the Create New QuickPatch Wizard.
- **Visual Basic .NET Wizard**—Select this option to launch the Visual Studio .NET Wizard for Visual Basic .NET, Visual C++ .NET, and C# .NET.
- **Visual C++ .NET Wizard**—Select this option to launch the Visual Studio .NET Wizard for Visual Basic .NET, Visual C++ .NET, and C# .NET.
- **C# .NET Wizard**—Select this option to launch the Visual Studio .NET Wizard for Visual Basic .NET, Visual C++ .NET, and C# .NET.

Express Projects

Express is the standard project type for creating installations in InstallShield. Express projects use the Windows Installer service to drive the entire installation. The flow of your installation's user interface (UI) is authored directly in the .msi package, and the Windows Installer service uses its user interface rendering capabilities to display the UI to your end users.

QuickPatch Projects

A QuickPatch project is a specific type of project recommended for installation authors who want to ship small, single updates to their users. Changes that are more extensive such as adding custom actions and changing .ini data typically require a standard patch.

QuickPatch authoring provides a simple alternative to creating a patch configuration in the Patch Design view, even though it provides less customization. Fundamentally, both patch creation methods produce the same deliverable types: .msp and .exe files.

With a QuickPatch, you can do any of the following:

- Add new files to the original installation or an earlier QuickPatch.

- Delete files in the original installation.
- Delete files that were added with a previous QuickPatch.
- Perform the same set of above operations on registry entries.
- Remove custom actions that were included with the original installation but that do not apply to the current QuickPatch project.

The creation of a QuickPatch project always begins with the Create New QuickPatch Wizard. Completing the wizard ensures that all basic requirements for the QuickPatch project are met. Then you can configure project settings once it opens in InstallShield.

Using Projects

InstallShield allows you to create, edit, upgrade, and save numerous project types—from installation projects to InstallShield object projects that allow you to reuse functionality within InstallShield.

The pages in this section cover a variety of topics, including how to create a particular project type, how to save projects, and related information.

Creating New Projects

There are a number of ways to create a new InstallShield project.

Task ***To create a new project, do one of the following:***

- Click the **New Project** button on the toolbar or on the **Start Page**.
- Press Ctrl+N.
- On the **File** menu, click **New**.

Any of these steps launches the New Project dialog box, where you can select the project that you want to create. If you select an installation project in the New Project dialog box, the Project Assistant launches to help you create your project.

Creating a Project From Within Microsoft Visual Studio

You can create an InstallShield project from within the Microsoft Visual Studio workspace. For more information, see Creating InstallShield Projects in Microsoft Visual Studio.

Opening Projects

Task ***To open an existing InstallShield project, do one of the following:***

- Click the **Open Project** button on the toolbar.
- On the **File** menu, click **Open**.

- Press CTRL+O.
- On the **Start Page** page, click the **Open an existing project** link or click a recently opened project link.
- Double-click a project file on the desktop or in Windows Explorer.

With the exception of clicking on a specific file or file link, all of the above options launch the Open dialog box, which allows you to browse to your project file.

If the installation project that you want to open was created with a version of InstallShield Express prior to version 3, see Upgrading from Earlier Versions of InstallShield.

Opening Projects Created with Earlier Versions of InstallShield

Task ***To open a project created with an earlier version of InstallShield:***

1. On the **File** menu, click **Open**. The **Open InstallShield Express Project** dialog box opens.
2. In the **Files of type** list, select the type of project that you would like to open.
3. Select the project file that you would like to open.
4. Click **Open** to begin migrating this project.

Saving Projects

When you create a new project, it is saved automatically with the name and location that you provide in the New Project dialog box.

InstallShield enables you to save a copy of the open project as a new project with a different name and location.

Saving a Project with a New Name and Location

When you save your project with a new name and location, a copy of the renamed project file and all of its associated files and folders are saved in the new location. The next time that you save your project, all changes are saved to this new location.

Task ***To save your project with a new name and location:***

1. On the **File** menu, click **Save As**. The **Save As** dialog box opens.
2. In the **Save in** box, select the appropriate location.

Note ▪ *When you save a project with external dependencies, your new project points to the original copies of these files. Duplicate copies are not made. Therefore, before you delete your original project, ensure that you do not delete any files that may be used by the new project.*

Changing the Default Project Location

All new projects are saved by default in the following location:

```
C:\InstallShield 2021 Projects
```

Task ***To specify a new default location for your setup projects:***

1. On the **Tools** menu, click **Options**. The **Options** dialog box opens.
2. Click the **File Locations** tab.
3. In the **Project Location** box, enter a new path or click **Browse** to find the appropriate location.
4. Click **OK**.

The default location for all new projects that you create with InstallShield is the one that you specified. Although this folder is used for all new projects, any existing projects remain in the previous location.

GUIDs

GUID stands for Globally Unique Identifier. A GUID is 128 bits long, and the algorithm used to generate a GUID guarantees each GUID to be unique. Because GUIDs are guaranteed to be unique, they can be used to identify COM classes, Product Codes, and various other codes.

For example, after a product is installed, a key is created under HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall and is named after the installation's Product Code. At one time, this key was named after the Product Name. However, this caused a potential conflict. If two installations were installed on the same machine, and both shared the same Product Name, they would also share the same registry key. Because a GUID is now used, it guarantees that this conflict does not occur.

An example GUID is {5D607F6A-AF48-4003-AFA8-69E019A4496F}. Any letters in a GUID must be in uppercase.

GUIDs in Your Project

When you create an installation project, there are a number of different GUIDs that are relevant to your project.

Table 2-3 ▪ GUIDs

GUID Name	Description
Product Code or Product GUID	The product GUID uniquely identifies your application.
Package Code or Package GUID	The package code uniquely identifies your installation package.
QuickPatch GUID	The patch GUID uniquely identifies a QuickPatch package.

Table 2-3 ▪ GUIDs (cont.)

GUID Name	Description
Upgrade Code or Upgrade GUID	The upgrade GUID uniquely identifies a family of products for upgrade purposes. It is important for upgrades.

For information on when you need to change GUIDs in your project, see Updating the Package Code, the Product Version, and the Product Code.

Sample Files

Some sample files are included with the InstallShield installation. These files are stored in the Samples folders in the following location:

InstallShield Program Files Folder\Samples

These files are used in conjunction with the tutorials that are available in the InstallShield Help Library.

Project Assistant

InstallShield includes a Project Assistant to help you quickly and easily build a basic installation project. The Project Assistant provides a framework of installation project tasks to guide you through the project creation process and provides pertinent information along the way.

How the Project Assistant Works

When you create a new installation project, the Project Assistant view automatically opens. The Project Assistant is available for the Express project type.

Information that you enter in the Project Assistant is saved directly to the underlying project file. Therefore, you can switch to the Installation Designer (described below) and view or modify your information using the full power of the InstallShield interface. In this way, you can use the Project Assistant to create the foundation for a more advanced installation where you use the Installation Designer, as needed, to customize your installation.

Integration with the Installation Designer

The Installation Designer tab, which is available for Express project types, displays all of the available views in the InstallShield interface. Here you can add more complex and powerful elements to your installation project. You can create your installation project using the Project Assistant and then use the Installation Designer to fine-tune project elements.

The Installation Designer and the Project Assistant run simultaneously. Any changes you make in one are reflected instantly in the other. For example, if you remove a feature while in the Installation Designer tab, that feature is no longer present in your installation project and does not appear in the Project Assistant.

Using the Project Assistant

When you create a new installation project, the Project Assistant is automatically activated. The Welcome page contains an installation design diagram to help you visualize the steps involved in creating an installation. You can work within the Project Assistant to create your project or click the Installation Designer tab to further define a basic installation project.

Using the List Controls

Each page contains up to three list controls, when applicable, to assist you in creating your installation and finding information:

- **More Options**—Provides additional configuration options relating to the specific area on the Project Assistant page. These are less common options that complete the functionality of the Project Assistant.
- **Other Places**—The view in the Installation Designer that corresponds to the current Project Assistant page. Clicking the link launches the full Installation Designer and activates that view.
- **Help Links**—This list provides links to help topics pertinent to the current Project Assistant page.

Navigating in the Project Assistant

Task ***To navigate from one page of the Project Assistant to another, do one of the following:***

- To navigate directly to a specific page, click the appropriate icon in the navigation bar at the bottom of the page.
- To follow the Project Assistant steps sequentially, do one of the following:
  - Click the Next or Back arrow buttons to move forward or backward.
  - Press CTRL+TAB to move to the next page and CTRL+SHIFT+TAB to move to the previous page.
- To move back to the Home page and view the Installation Design Diagram, click the Home button on the navigation bar.

Opening the Installation Designer

The Installation Designer tab displays the views in the InstallShield Installation Development Environment (IDE). Here you can add more complex and powerful elements to your installation project. To open a view in the Installation Designer, click the Installation Designer tab.

Note ▪ *The Installation Designer and the Project Assistant run simultaneously. Any changes you make in one are reflected instantly in the other.*

Hiding the Project Assistant

Task ***To hide the Project Assistant:***

On the **View** menu, click **Project Assistant**.

The check mark next to the Project Assistant command is removed, and the Project Assistant is hidden.

The Project Assistant remains hidden until it is selected in the View menu. When you create a new project and the Project Assistant command does not have a check mark, the Installation Designer becomes the default tab.

Application Information Page

The Application Information page is where you specify general information about the application your project will install, including the application's name and version, your company's name and Web address, and the application icon.

Add or Remove Programs in the Control Panel

The Add or Remove Programs in the Control Panel provides a list of the applications that are installed on a computer system. You can view information about particular programs and add, modify, or remove programs.

The information you provide in the Application Information page of the Project Assistant is used to populate information for your application's Add or Remove Programs information when your application is installed.

Company Name and Product Name in Your Installation

Your company name and product name are used in several places in your installation project.

How Your Company Name Is Used Throughout Your Installation Project

Your company name is used to set the default installation directory for your application. It is also used in Add or Remove Programs in the Control Panel for your application on the end user's system.

How Your Product Name Is Used Throughout the Installation Project

Your product name is used in your application's entry in Add or Remove Programs (in the support information link) on the target system. It is also used in setting the default installation directory.

Installation Requirements Page

The Installation Requirements page allows you to easily set installation requirements for the target system. For example, if your application requires a specific operating system in order to run properly, you can indicate that in the first section of this page.

Specifying Operating System Requirements in the Project Assistant

When you specify operating system requirements on the Installation Requirements page of the Project Assistant, InstallShield creates launch conditions. These conditions are added to the `LaunchCondition` table of your .msi file.

How InstallShield Creates the Operating System Launch Conditions

When you specify operating system requirements on the Installation Requirements page, you are essentially excluding operating systems that do not support your application.

For example, if you select only the check box for the latest Windows operating system, InstallShield creates a launch condition to exclude the operating systems that you did not select on the Installation Requirements page. With this type of launch condition, future versions of Windows operating systems are supported automatically because they are not excluded in the launch condition.

When Does the Installation Check for Requirements?

To ensure that the required software or operating system is present on the target system, the installation checks for these requirements in the beginning of the installation before any files are transferred.

Modifying the Run-Time Message for Software Requirements

If your installation has software requirements and the target system does not have the selected software, a run-time message is displayed during the installation. You can modify the message that is displayed.

Task ***To modify the run-time message:***

1. In the Project Assistant, open the **Installation Requirements** page.
2. Select **Yes** in answer to the software requirements question.
3. Select the software that your application requires. The default run-time message is displayed to the right.
4. Click the run-time message to edit it.

Creating Custom Installation Requirements

You can specify target system requirements in your installation project by using the Project Assistant or the Requirements view of the Installation Designer (IDE). When you specify system software requirements, you must run through the System Search Wizard. The System Search Wizard provides the Windows Installer capability to search for a particular file, folder, registry key or .ini value on a target system prior to installation.

Task ***To access the System Search Wizard in the Project Assistant:***

1. In the Project Assistant, open the **Installation Requirements** page.
2. In the **More Options** area, click **Create a custom software condition**.

Installation Architecture Page

The Installation Architecture page lets you specify the features that you want your installation program to display to the end user. A feature is the smallest separately installable piece of your product from the end user's standpoint. Individual features are visible to end users when they select a Custom setup type during installation.

***Note** ▪ Features can contain subfeatures, subsubfeatures, and so forth, to as many levels as your installation program requires.*

Adding Features in the Project Assistant

Task ***To add a feature:***

1. In the Project Assistant, open the **Installation Architecture** page.
2. Select **Yes** in answer to **Do you want to customize your Installation Architecture?**
3. To add a main feature, click the **Installation Architecture** explorer. To add a subfeature, click the feature that you want to be the parent feature, and then click **New**. The Project Assistant creates the new feature.
4. Name the feature or click **Rename** to name it later.

Determining Whether to Create a Multiple-Feature Installation

Features are the building blocks of an installation, from the end user's perspective. Because of this, features should represent distinct and discrete pieces of functionality within your installation.

If your application has different blocks of functionality, you should create a multiple-feature installation. For example, if your installation contains your application (.exe file) and a help library (.hlp file), your installation project should contain at least two features—one for each piece of functionality.

To learn more about creating a multiple-feature installation within the Project Assistant, see Creating Installations with Multiple Features.

Creating Installations with Multiple Features

You can create an installation with multiple features using the Project Assistant or in the Installation Designer (IDE).

Task ***To create a multiple-feature installation in the Project Assistant:***

1. In the Project Assistant, open the **Installation Architecture** page.
2. Select **Yes** to indicate you want to customize your installation architecture.
3. Click the **Installation Architecture** explorer and then click **New**. The Project Assistant creates a new feature.

4. Press F2 or right-click the feature and select **Rename** to provide a name for the new feature.
5. To add another feature at the same level, click the **Installation Architecture** explorer and then click **New**. To create a subfeature, click the feature that you want to be the parent feature, and then click **New**. The Project Assistant creates the new feature.
6. Continue adding features and subfeatures as needed.

Default Features

The *default feature* concept exists only in the Project Assistant. All resources (for example, files or registry data) that are added to an installation project need to be associated with a feature. If a resource is not associated with a feature, it is not installed to the target system at run time.

Using a default feature simplifies the authoring experience in the Project Assistant. You do not need to worry about associating project resources with a feature to ensure that they are installed. When you add registry data, create new shortcuts, or add files when All Application Data is selected, all of these resources are added to the default feature. This ensures that all of the project resources you add in the Project Assistant will be installed to the target system when an end user runs your installation.

Setting the Default Feature

You can set the default feature in the Installation Architecture page of the Project Assistant.

What Happens If There Are No Features or No Default Feature Is Selected?

When you navigate to the Installation Architecture page or add data to the Application Files, Application Shortcuts, or Application Registry page, InstallShield selects the first root feature as the default feature. If there are no features, InstallShield creates one silently.

Defining Feature Hierarchy

Top-level features are the highest level in the feature hierarchy. Top-level features might include the application you want to install, a help library feature, and a sample projects feature.

Beneath the top-level features are subfeatures or child features. This is a feature that is dependent upon another feature for installation purposes. If the parent (or top-level) feature is not installed to the target system, the child feature is not installed.

Application Files Page

The Application Files page lets you specify the files you want to associate with each of your features.

Adding Files to Features in the Project Assistant

Task ***To add a file to a feature:***

1. In the Project Assistant, open the **Application Files** page.
2. In the feature list at the top of the page, select the feature that should contain the file.
3. In **Destination Computer** explorer, select the folder to which you want to add the file.
4. Click **Add Files**. The **Open** dialog box opens.
5. Browse to the file that you want to add.
6. Click **Open** to add the file to the selected feature. The “The file you have added ... may have dependencies” message appears.
7. Click **Yes** if you want to have those dependencies automatically added to your installation project.

Removing Files from Features in the Project Assistant

Task ***To remove a file from a feature:***

1. In the Project Assistant, open the **Application Files** page.
2. Click the file you want to remove and press **Delete**.

Adding Files to a Fixed Folder Location

If you know exactly where you want the installation to install your project files on the target system, you can hard-code a fixed folder destination.

Task ***To add files to a fixed folder location in the Project Assistant:***

1. In the Project Assistant, open the **Application Files** page.
2. Right-click **Destination Computer** and click **New Folder**.
3. For the new folder’s name, type the drive in which the destination is located—for example, **C:**.
4. Beneath the drive letter folder, further define the destination path by adding subfolders.

Viewing Additional Predefined Folders

The Project Assistant’s Application Files page displays the more commonly used predefined folders. You can view and hide predefined folders in this page.

Task ***To display additional predefined folders:***

1. In the Project Assistant, open the **Application Files** page.
2. Right-click **Destination Computer** and click **Show Predefined Folder**.
3. In the list of predefined folders, select the folder you want to display.

Tip ▪ *To hide predefined folders, deselect them in the list of predefined folders.*

Application Shortcuts Page

The Application Shortcuts page lets you specify shortcuts for your application's files on the target system's desktop or Start menu. By default, this page displays a shortcut for each executable file that you have added to your project through the Project Assistant. You can delete these, and add shortcuts to other files that you have included in your installation project.

File Extensions

File name–extension associations, or file associations, are registry settings that tell Windows what application to use to open files of a certain type. For example, Windows typically opens text files (files with the .txt extension) with Windows Notepad and opens bitmap files (files with the .bmp extension) with Microsoft Paint.

A file extension allows someone to identify the type of file without accessing the file. A suffix (**.abc**) is appended to the file name. The file extension is also useful because another application can recognize whether the application can work with the file (for example, open the file or modify it), based on the extension.

In InstallShield, you can register your own file extensions in the File Extensions view. Registering your file extension instructs the target machine's operating system to use your application to open files with your file extension when an end user opens a file.

Task ***To access the File Extensions view in the Project Assistant:***

On the **Application Shortcuts** page, in the **Other Options** area, click **File Extensions**. The **File Extensions** view opens.

Creating Shortcuts to Files That Are Not Included in the Installation

You can configure your installation to create a shortcut that points to a file that already exists on the target system. This file does not have to be included in your installation project.

Task ***To create a shortcut to a file that is not included in the installation:***

1. Open the Installation Designer.
2. In the View List under **Configure the Target System**, click **Shortcuts/Folders**.
3. In the **Shortcuts** explorer, right-click the destination directory that you want to contain the shortcut and then click **New Shortcut to Preexisting File**. The **Browse for Shortcut Target** dialog opens.
4. Browse to the target file's location and enter the file's name in the **File Name** setting.
5. Click **OK**.
6. Configure the shortcut's settings.

Modifying a Default Shortcut in the Project Assistant

Task ***To modify a default shortcut:***

1. In the **Project Assistant**, open the **Application Shortcuts** page.
2. Select the shortcut that you want to modify.
3. Make the required changes as needed.

Associating a Shortcut's Target with a File Extension in the Project Assistant

You can associate a shortcut's target with a file extension. When you do this, Windows will use the target file to launch files with the specified file extension. For example, if you enter .txt, when the end user opens a .txt file, the target file of this shortcut is launched to open the .txt file.

Task ***To associate a shortcut's target with a file extension:***

1. In the **Project Assistant**, open the **Application Shortcuts** page.
2. Click the shortcut to activate the shortcut options.
3. Select the **Associate shortcut with file extension** option.
4. Type the file extension that you want to associate with this shortcut's target—for example, **txt**. You can add multiple extensions by separating them with a comma.

Application Registry Page

The Application Registry page lets you specify any registry data that your application requires.

Updating the Registry

The registry is a database for your computer's configuration information. Information included in a computer's registry includes user profiles, hardware and software installed on the computer, and property settings.

How Do I Know What Registry Data My Application Requires?

The application developer should be able to provide registry information for you. Specifically, you will need to know if the application you are installing requires any user-specific (HKEY_CURRENT_USER) or machine-specific (HKEY_LOCAL_MACHINE) settings.

The developer can provide a .reg file that you add to your installation. InstallShield allows you to import .reg files into your installation project.

Configuring Registry Data in the Project Assistant

Task ***To configure registry data:***

1. In the **Project Assistant**, open the **Application Registry** page.
2. Select **Yes** in answer to the question about configuring registry data.
3. Right-click the registry item to which you want to add the data, select **New**, and point to **Key**.
4. Name the key.
5. Right-click the key, select **New**, and point to the appropriate command. You can pick Default Value, String Value, Binary Value, DWORD Value, Multi-String Value, or Expandable String Value, depending on the type of data you want to register.

Modifying Registry Data Values in the Project Assistant

Task ***To modify registry data:***

1. In the **Project Assistant**, open the **Application Registry** page.
2. Double-click the data. The **Edit** dialog box opens.
3. Edit the data and click **OK**.

Associating Registry Data with Features

For Express projects, all of the registry data that you add on the Project Assistant's Application Registry page is added to your project's default feature. You can associate your registry data with another feature in the Installation Designer.

Task ***To use the Installation Designer to associate registry data with a feature other than the default feature:***

1. In the View List under **Configure the Target System**, click **Registry**.
2. In the **View Filter** list, select the feature with which you want to associate the registry data.
3. Create or drag and drop the registry data in the appropriate registry location.

Using Variable Data Types in Registry Data

Project ▪ *This information applies to Express projects.*

InstallShield allows you to use variable data types or properties when creating registry data for your installation project.

Task ***To use INSTALLDIR as a variable in the registry:***

1. In the **Project Assistant**, open the **Application Registry** page.
2. Select **Yes** to indicate that you want to configure the registry data that your application will install.
3. Right-click **HKEY_CLASSES_ROOT**, point to **New**, and select **Key**.
4. Name the key **Installation Location**.
5. Right-click the **Installation Location** key, point to **New**, and select **String Value.**
6. Name the string value **My Installation Location**.
7. Double-click the **My Installation Location** key. The **Edit Data** dialog box opens.
8. In the **Value Data** field, type `[INSTALLDIR]`.

At run time, the value of `[INSTALLDIR]` is replaced with the installation directory.

Application Paths

The application path registry key contains data that Windows uses as a private search path for the specified application's .dll files. If you install an application's .dll files into a directory not found in the PATH environment variable (and not into the application's directory), you should set the appropriate application path to include the .dll file directory during installation. Application path information is stored in the registry under HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\App Paths\AppName.exe.

Installation Interview Page

The Installation Interview page lets you specify the dialogs that you want end users to see when your installation program runs. Based on your answers to the questions on this page, the Project Assistant adds the corresponding dialog box to your installation project.

Specifying Dialogs for Your Installation in the Project Assistant

The questions that are displayed on the Installation Interview page of the Project Assistant help you specify which dialogs you want the end user to see when your installation program runs.

Task ***To specify dialogs for your installation:***

1. **Do you want to display a License Agreement Dialog?—**Select **Yes** to browse to your license agreement file.
2. **Do you want to prompt users to enter their company name and user name?—**Select **Yes** to display a dialog requesting this information.
3. **Do you want your users to be prompted to modify the installation location of your application?**—Select **Yes** to present a dialog that allows end users to change the installation location. See Allowing End Users to Modify the Installation Location for more information.
4. **Do you want users to be able to selectively install only certain parts of your application?—**Select **Yes** to allow end users to select which parts of the application they want to install. See Creating Selectively Installable Installations for more information.
5. **Do you want to give users the option to launch your application when the installation completes?—**Select **Yes** and browse to your application file. When this option is set to **Yes**, the final dialog in the installation presents a check box that allows end users to immediately launch your application upon clicking the **Finish** button.

Tip ▪ *To add a custom graphic to your installation dialogs, click the link in the More Options section of this page to launch the Dialog Images dialog box.*

Allowing End Users to Modify the Installation Location

If you want to provide end users with control over where your software is installed on their system, you can allow them to modify the installation location.

The Windows Installer property `INSTALLDIR` serves as the default installation directory. When you allow users to modify the installation location, the Destination Folder dialog is presented during the installation.

License Agreements

To install your application, end users must agree to abide by certain legal requirements. For example, most software vendors do not allow users to copy or distribute their software to others.

To ensure that the end user understands the legal requirements associated with installing your software, your installation can present an End-User License Agreement (EULA) in the License Agreement dialog during run time. The EULA is a legal contract between you and the end user, with regard to the use of your software.

The License Agreement dialog displays your license agreement text and contains Yes and No options. If the end user does not agree to accept the EULA, your software is not installed and the installation terminates.

Task ***To add a License Agreement dialog to your project in the Project Assistant:***

1. In the **Project Assistant**, open the **Installation Interview** page.
2. Select **Yes** to indicate that you want to add a **License Agreement** dialog.
3. Type the path to your license agreement file or browse to the file. The file must be a rich text format (.rtf) file.

Creating Selectively Installable Installations

You can allow your end users to select which portions of your installation they want to install to their systems. This is a custom installation, which provides a list of the available features within your installation. The end user can select the features to install in a dialog presented at run time.

For example, your installation might contain your application's executable (.exe) file, a documentation (.chm) file, and a samples file. All of these files are contained in different features, which are provided in an option list to the end user. If the end user needs only the application, they can select to install the executable file, but not the documentation or samples file.

Build Installation Page

Tip ▪ *The following instructions apply to releases that are built within InstallShield (without integration with Visual Studio). For information on building an InstallShield release from within Visual Studio, see Building Releases in Microsoft Visual Studio.*

The Build Installation page lets you specify what type of distribution you want to build and, optionally, the location to which you want to copy the distribution files. It also enables you to digitally sign the package.

Building Your Installation from the Project Assistant

Tip ▪ *The following instructions apply to releases that are built within InstallShield (without integration with Visual Studio). For information on building an InstallShield release from within Visual Studio, see Building Releases in Microsoft Visual Studio.*

Task ***To build your installation:***

1. In the **Project Assistant**, open the **Build Installation** page.
2. Select the installation types that you want to build.
3. If you want InstallShield to automatically copy your installation to another location after the build finishes, click the **Optional distribution settings** link for each build option and specify the location.

   If you want InstallShield to distribute your installation after the build finishes, click the **Optional distribution settings** link for each build option and select the **Distribute After Build** check box.

4. To digitally sign your `Setup.exe` file to assure your end users that the code within your application has not been modified or corrupted, click the **Digitally Sign Setup hyperlink**. The **Digitally Sign Setup** dialog box opens. Configure the settings as needed.
5. Click **Build Installations**.

The Output window opens; it displays information about the progress of the build. The build is finished when the Output tab displays the log file information.

After Completing the Project Assistant: Next Steps

After going through the Project Assistant pages and completing the fields, you have an installation project framework that you can use as a functional installation, or you can further customize to fit your needs.

Further Customizing Your Project

The Installation Designer provides an easy way to access all of the installation creation views available in InstallShield. Click the Installation Designer tab at the top of the Project Assistant workspace to display the views.

Within the Installation Designer, you can see the View List, which is a list of all available views for the project type that you are creating. To display the View List on the left side of the workspace, press F4.

Learning About InstallShield Views

For detailed information about each of the InstallShield views, refer to View Reference.

Working with the InstallShield Interface

The InstallShield interface is a graphical user interface with conventional Windows-based elements such as a menu bar, a toolbar, and dialog boxes. This section includes topics that explain how to perform basic tasks using these elements and how to customize the interface.

Displaying the View List

Task ***To see the View List in the InstallShield interface:***

1. Click the **Installation Designer** tab at the top of the InstallShield interface.
2. Do one of the following to display the View List:
   - On the **View** menu, click **View List**. The **View List** appears on the left side of the InstallShield interface.
   - Click the toolbar's **View List** button.
   - Press F4 to hide or show the **View List**.

Opening Views in the InstallShield User Interface

The first step in many InstallShield procedures is to open a particular view in the Installation Development Environment (IDE).

Task ***To open a view:***

1. Click the **Installation Designer** tab. The **View List** is displayed along the left side of the IDE. If the **View List** is not displayed, see Displaying the View List.
2. In the **View List**, select the view you want to open. To see all available views, expand the View List folders.

Working with the Group Box Area in Various Views

A number of views in InstallShield have a group box area that lets you organize the rows in the view. The views that contain this group box support multiple levels of grouping simply by dragging the column headers and dropping them onto the group box (the area that says, “Drag a column header here to group that column”). InstallShield displays the rows in the view hierarchically according to column arrangement in the group box. An example of a view that contains the group box is the Redistributables view.

Note the following tips for working with the group box:

- To move a column header to the group box area, drag the column header and drop it onto the group box.
- To copy a column header onto the group box area, press CTRL while dragging the column header and dropping it onto the group box. When you do this, the column header is displayed as a column header, and in the group box.
- When you are dropping group box headers onto the group box area, you can drop them onto other column headers. This enables you to organize rows hierarchically.
- To remove a column header from the group box, drag it from the group box area and drop it onto the row of column headers. When you are dragging it over the row of column headers, InstallShield displays arrows to indicate where in the row the column header would be displayed if you dropped it.
- To sort the items in the grid by a particular column header, click the column header in the group box or in the row of column headers.

The following examples demonstrate different ways for using the group box to organize content in views.

Default Behavior: Empty Group Box Area

By default, no column headers are displayed in the group box. The following screen shot shows part of the Redistributables view. Items are sorted by the Name column.

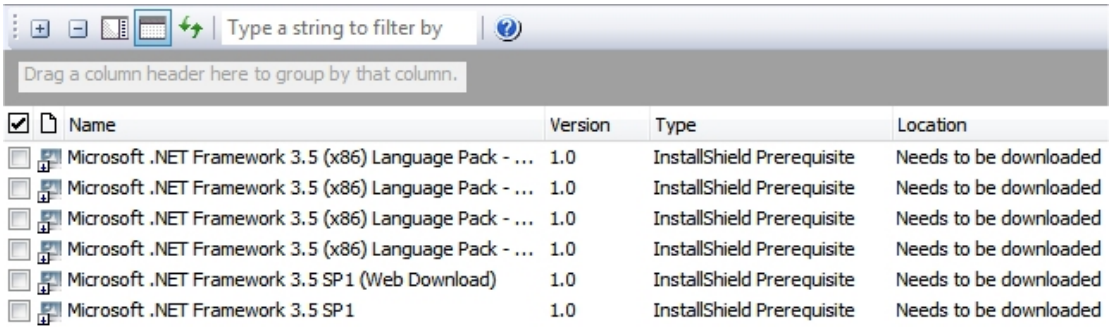


Figure 2-3: Empty Group Box Area

Grouping by One Column Header

If you press CTRL while dragging and dropping one column header onto the group box, InstallShield arranges the rows in the grid into groups of items. The following screen shot shows part of the Redistributables view. Rows are organized to help identify the redistributables that have been added to the project (that is, the redistributables whose check box is selected).

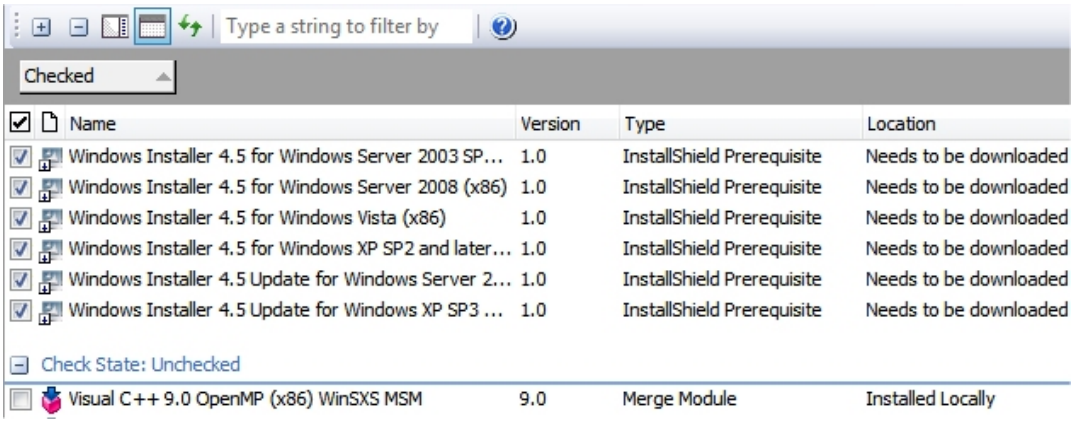


Figure 2-4: Grouping Rows by One Column Header

Grouping by Two Column Headers

If you press CTRL while dragging and dropping one column header onto another column header in the group box, InstallShield arranges the rows in the grid into multiple groups of items. The following screen shot shows part of the Redistributables view. Rows are organized to help identify the InstallShield prerequisites and merge modules that have been added to the project.

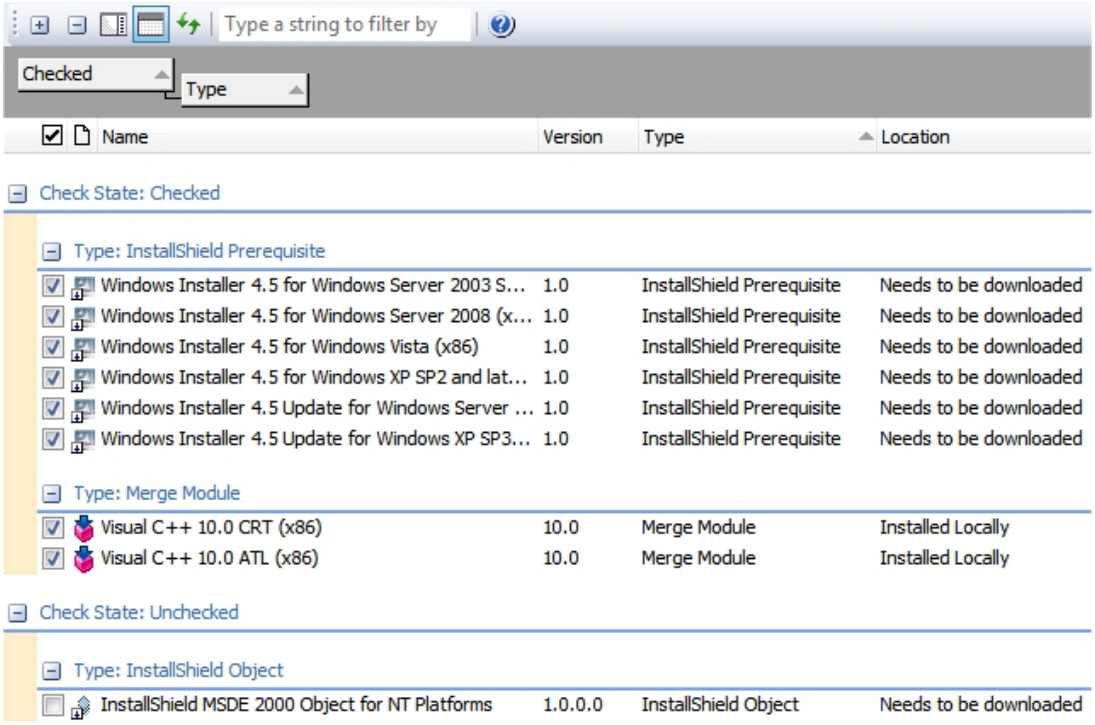


Figure 2-5: Sorting Rows by the Check Box Column and Then the Type Column

Showing or Hiding Toolbars

Task ***To show or hide a toolbar, do one of the following:***

- Right-click a toolbar and select the toolbar that you want to be displayed or hidden.
- On the **Tools** menu, click **Customize**. The **Customize** dialog box opens. Select the check box for each toolbar that you want to be displayed. Clear the check box for each toolbar that you want to be hidden.

Adding Buttons and Menus to a Toolbar

Task ***To add a button or menu to a toolbar:***

1. Ensure that the toolbar that you want to change is visible.
2. On the **Tools** menu, click **Customize**. The **Customize** dialog box opens.
3. Click the **Commands** tab.
4. In the **Categories** box, click the category for the button or menu that you want to add.
5. Drag the button or menu from the **Commands** box to the appropriate toolbar.

Tip ▪ *To create your own custom toolbar, drag the button or menu to the empty gray area near the toolbars.*

Removing Buttons and Menus from a Toolbar

Task ***To remove a button or menu from a toolbar:***

1. Ensure that the toolbar that you want to change is visible.
2. On the **Tools** menu, click **Customize**. The **Customize** dialog box opens.
3. Right-click the button or menu that you want to remove, and then click **Delete**.

Creating Custom Toolbars

Task ***To create a custom toolbar:***

1. On the **Tools** menu, click **Customize**. The **Customize** dialog box opens.
2. Click the **Toolbars** tab.
3. Click the **New** button. The **New Toolbar** dialog box opens.
4. In the **Toolbar name** box, enter a descriptive name for the toolbar, and click **OK**.
5. Customize the new toolbar by adding menus or buttons.

Docking or Undocking the Output Window

The Output window or its individual tabs can be docked to any side of the workspace in InstallShield, or they can be dragged to free-floating positions.

If you drag the Output window or one of its tabs to the edge of the InstallShield interface, it becomes a docked window. If you drag the Output window or one of its tabs away from any of the edges of the InstallShield interface, it becomes undocked.

Task ***To undock the Output window:***

Drag the title bar of the Output window to the new location. Resize the Output window as needed.

Task ***To dock the Output window:***

Drag the title bar of the Output window to the left, right, top, or bottom edge of the InstallShield interface.

Task ***To undock a tab on the Output window:***

Drag the tab to the new location. Resize the Output window as needed.

Task ***To dock one of the Output window tabs:***

Drag the tab to the left, right, top, or bottom edge of the InstallShield interface.

Configuring Advanced Settings for InstallShield

One of the InstallShield program files is a file called `Settings.xml`. This file exposes some advanced machine-wide settings for InstallShield. When you install InstallShield, `Settings.xml` is installed to one of the following locations, depending on which language version of InstallShield you are using:

- **English**—*`InstallShield Program Files Folder`*\Support\0409
- **Japanese**—*`InstallShield Program Files Folder`*\Support\0411

In most cases, you should not modify the `Settings.xml` file. However, in some cases, you may need to make changes in this file. This section of the documentation describes some scenarios that would require `Settings.xml` changes:

- Changing the Timestamp Server for Digital Signatures
- Configuring the Compression Level for Files that Are Streamed into Setup.exe
- Configuring the Maximum Size for .cab Files
- Configuring Platform Architecture for Digital Signing

Caution ▪ *The* `Settings.xml` *file contains critical data; if this file is edited incorrectly, it can cause InstallShield to fail to work. Use extreme care when editing this file.*

Changing the Timestamp Server for Digital Signatures

Project ▪ *This information applies to the following project types:*

- *Express*
- *QuickPatch*

When you specify digital signature information for a release, InstallShield uses VeriSign's server (**http://timestamp.verisign.com/scripts/timstamp.dll**) as the default timestamp server during builds. InstallShield includes a machine-wide setting that lets you replace that default server with a different timestamp server. The setting also lets you disable timestamping.

InstallShield 2019 R2 and later Express Edition supports adding a delay between the successive digital signing, this requires only if the timestamp server fails handle the successive signing requests.

Required to specify the <DelayBetweenSigning default="1500"/> node in the settings.xml, under <DevStudio/Build> node in Settings.xml in milliseconds.

Caution ▪ *The following instructions require that you modify the* `Settings.xml` *file that is installed with InstallShield. This file contains critical data; if it is edited incorrectly, it can cause InstallShield to fail to work. Use extreme care when editing this file.*

Task ***To configure the timestamp server for digital signatures:***

1. Close InstallShield.
2. Find the `Settings.xml` file that is installed with InstallShield. `Settings.xml` is installed in one of the following locations, depending on which language version of InstallShield you are using:
   - **English**—*`InstallShield Program Files Folder`*\Support\0409
   - **Japanese**—*`InstallShield Program Files Folder`*\Support\0411
3. Create a back-up copy of the `Settings.xml` file, in case you later need to revert to the original version.
4. Use a text editor or XML file editor to open the `Settings.xml` file.
5. Search for the <DigitalSignature> element. It looks something like this:

   ```
   <DigitalSignature Timestamp="http://timestamp.verisign.com/scripts/timstamp.dll"/>
   ```

6. To override the timestamp server with a different one, set the value of the Timestamp attribute to the appropriate URL.

   To disable timestamping, use an empty value for the Timestamp attribute:

   ```
   <DigitalSignature Timestamp=""/>
   ```

   Note ▪ *Disabling timestamping may affect how long your digital signature is considered to be valid.*
7. Save the `Settings.xml` file.
8. Ensure that your XML code is well formed; if it is not well formed, you may have problems using InstallShield. In most cases, you can identify improperly formed XML code by opening the `Settings.xml` file in Internet Explorer. You should be able to expand and contract the major elements of the file; if you cannot, check the code for errors.

Whenever you build a release that includes digital signature information, InstallShield sets the timestamp according to the setting that you configured.

Configuring the Compression Level for Files that Are Streamed into Setup.exe

InstallShield includes a machine-wide setting that lets you specify the compression level that you want to use for files that are streamed into `Setup.exe` files at build time. Following are examples of files that may be streamed into the `Setup.exe` files:

- All of your product's files (if the release is one in which all of the files are compressed into the `Setup.exe` setup launcher)
- InstallShield prerequisite installations that have a location of Extract From Setup.exe
- The .NET Framework installation if it has a location of Extract From Setup.exe
- The Windows Installer installation if it has a location of Extract Engine From Setup.exe

InstallShield also lets you specify particular files (with or without wild-card characters) that should not be compressed when they are streamed into `Setup.exe` files.

Caution ▪ *The following instructions require that you modify the* `Settings.xml` *file that is installed with InstallShield. This file contains critical data; if it is edited incorrectly, it can cause InstallShield to fail to work. Use extreme care when editing this file.*

Task ***To configure compression settings for streamed files:***

1. Close InstallShield.
2. Find the `Settings.xml` file that is installed with InstallShield. `Settings.xml` is installed in one of the following locations, depending on which language version of InstallShield you are using:
   - **English**—*`InstallShield Program Files Folder`*\Support\0409
   - **Japanese**—*`InstallShield Program Files Folder`*\Support\0411
3. Create a back-up copy of the `Settings.xml`, in case you later need to revert to the original version.
4. Use a text editor or XML file editor to open the `Settings.xml` file.
5. Search for the <StreamCompression> element. It looks something like this:

   ```
   <StreamCompression exclude="*.CAB" compressionlevel="-1"/>
   ```

6. To prevent certain files or file types from being compressed when they are streamed into `Setup.exe` files, set the value of the exclude attribute to the names of those files. Note the following guidelines:
   - To specify more than one file, separate each file name with a comma.
   - To indicate a wild-card character, use an asterisk (*).

   For example, to specify that .cab files, .exe files, and a file called **test.txt** should not be compressed, set the value of the exclude attribute to as follows:

   ```
   <StreamCompression exclude="*.CAB,*EXE,test.txt" compressionlevel="-1"/>
   ```

   The default value is ***.CAB**, since .cab files are a compressed type of file.

7. Do one of the following:
   - If you want to use a compression level that offers a balance between the size of the compressed file and the time that is required to extract the compressed files at run time, set the value of the compressionlevel attribute to `-1`. This is the default value.
   - To specify a particular compression level, set the value of the compressionlevel attribute to a number from 0 to 9, where 0 indicates no compression and 9 indicates maximum compression.

     In general, when you specify a number from 0 to 9, the higher the number that you specify, the smaller the resulting compressed file is, and the longer it may take to extract the files at run time.
8. Save the `Settings.xml` file.
9. Ensure that your XML code is well formed; if it is not well formed, you may have problems using InstallShield. In most cases, you can identify improperly formed XML code by opening the `Settings.xml` file in Internet Explorer. You should be able to expand and contract the major elements of the file; if you cannot, check the code for errors.

Whenever you build a compressed release, InstallShield streams files into `Setup.exe` files according to the settings that you configured.

Configuring the Maximum Size for .cab Files

Note ▪ *This information is applicable only if you are building a compressed SingleImage release in which all of the files are embedded in the single-file .msi package or the* `Setup.exe` *setup launcher.*

The .cab file type has some limitations. For example, the maximum size of a single .cab file is 2 GB. In addition, some users have had trouble signing large .cab files and verifying the digital signature of large signed .cab files.

To work around these limitations, InstallShield enables you to specify on a machine-wide basis the maximum size for each .cab file that is built for a compressed SingleImage type of release. When InstallShield is creating the .cab files for your release and it reaches the .cab file threshold that you configured, it splits the data into two or more .cab files, creating multi-part .cab files. Note that if you do not want InstallShield to create multi-part .cab files, you can configure InstallShield to store the data in a single .cab file.

Caution ▪ *The following instructions require that you modify the* `Settings.xml` *file that is installed with InstallShield. This file contains critical data; if it is edited incorrectly, it can cause InstallShield to fail to work. Use extreme care when editing this file.*

Task ***To specify whether InstallShield should create multi-part .cab files and—if appropriate—specify the maximum .cab file size:***

1. Close InstallShield.
2. Find the `Settings.xml` file that is installed with InstallShield. `Settings.xml` is installed in one of the following locations, depending on which language version of InstallShield you are using:
   - **English**—*`InstallShield Program Files Folder`*\Support\0409

- **Japanese**—*InstallShield Program Files Folder*\Support\0411

3. Create a back-up copy of the `Settings.xml`, in case you later need to revert to the original version.
4. Use a text editor or XML file editor to open the `Settings.xml` file.
5. Search for the <CompressedNetworkCABSize> element. It looks something like this:

   ```
   <CompressedNetworkCABSize default="600"/>
   ```

6. Do one of the following:

   - To specify the maximum size for .cab files, type the size in MB as the value of the default attribute. In the aforementioned example, the maximum size is **600**. The value should be 2048 or less, since the maximum size of a .cab file is 2 GB (2048 MB).

     The default value is **`600`**.

   - If you do not want InstallShield to create multi-part .cab files, set the value of the default attribute to **`-1`**.

7. Save the `Settings.xml` file.
8. Ensure that your XML code is well formed; if it is not well formed, you may have problems using InstallShield. In most cases, you can identify improperly formed XML code by opening the `Settings.xml` file in Internet Explorer. You should be able to expand and contract the major elements of the file; if you cannot, check the code for errors.

Whenever you build a compressed SingleImage release for one of the applicable project types, InstallShield creates the .cab files according to the requirement that you configured in the `Settings.xml` file. Depending on the value that you specified in the `Settings.xml` file, the `Media` table of the .msi package lists one or more .cab files.

Configuring Platform Architecture for Digital Signing

You can select platform architecture for digital signing of files.

Task ***To configure platform architecture for digital signing:***

1. Close **InstallShield**.
2. Find the **Settings.xml** file that is installed with InstallShield. **Settings.xml** is installed in one of the following locations, depending on which language of InstallShield that you are using:

   - **English**—*InstallShield Program Files Folder*\Support\0409
   - **Japanese**—*InstallShield Program Files Folder*\Support\0411

3. Create a backup copy of the **Settings.xml** file, in case you later need to revert to the original version.
4. Use a text editor or XML file editor to open **Settings.xml** file.

   Note ▪ *The file should be opened in Administrator mode for modification.*

5. Search for the tag **<DigitalSignature>**. The actual tag is **<DigitalSignature Platform="X86"/>.**
6. To use a different platform architecture, set Platform to either X86 or X64.
   For example,
   - To use 32-bit Signing: <DigitalSignature Platform="X86"/>
   - To use 64-bit Signing: <DigitalSignature Platform="X64"/>

   Note ▪ *Invalid entry will automatically use 32-bit Signing.*
7. Save the **Settings.xml** file.

Note ▪ *You cannot use 64-bit signing on a 32-bit Operating System.*

Upgrading from Earlier Versions of InstallShield

If you have setup projects that were created with earlier versions of InstallShield or InstallShield Express, you can open those projects with your new version of InstallShield. Because of the underlying technical differences between earlier versions and InstallShield 2026 Express Edition, the upgrade process does not always produce a one-to-one correlation between what was in your old project and what makes it into the new project. For example, InstallShield objects, which existed in earlier versions of InstallShield Express, have since been replaced by merge modules. In some cases, what may have been two objects in your old project could be replaced by a single merge module in your new project.

To learn more about migrating from earlier versions of InstallShield, see this section of the documentation.

- Upgrading Projects from InstallShield 2014 Express Edition or Earlier
- Upgrading Projects from InstallShield 2013 Express Edition or Earlier
- Upgrading Projects from InstallShield 2012 Spring Express Edition or Earlier
- Upgrading Projects from InstallShield 2012 Express Edition or Earlier
- Upgrading Projects from InstallShield 2011 Express Edition or Earlier
- Upgrading Projects from InstallShield 2010 Express Edition or Earlier
- Upgrading Projects from InstallShield 2009 Express Edition or Earlier
- Upgrading Projects from InstallShield 2008 Express Edition or Earlier
- Upgrading Projects from InstallShield 12 Express Edition or Earlier
- Upgrading Projects from InstallShield Express 2.x

Upgrading Projects from InstallShield 2014 Express Edition or Earlier

The following information describes possible upgrade issues that may occur when you upgrade projects that were created with InstallShield 2014 Express Edition and earlier to InstallShield 2026 Express Edition. It also alerts you to possible changes in behavior that you may notice between new InstallShield 2026 Express Edition projects and projects that are upgraded from InstallShield 2014 Express Edition or earlier to InstallShield 2026 Express Edition.

General Information about Upgrading Projects that Were Created in Earlier Versions of InstallShield

If you use InstallShield 2026 Express Edition to open a project that was created with an earlier version, InstallShield 2026 Express Edition displays a message box that asks you if you want to convert the project to the new version. If you reply that you do want to convert it, InstallShield creates a backup copy of the project with a file extension such as .775 before converting it. Delete the .775 part from the original project's file name if you want to reopen the project in the earlier version of InstallShield. Note that you cannot open InstallShield 2026 Express Edition projects in earlier versions of InstallShield.

You can upgrade projects that were created with the following versions of InstallShield Express Edition to InstallShield 2026 Express Edition: InstallShield 2014 Express Edition and earlier, InstallShield 12 Express Edition and earlier, and InstallShield Express 5 and earlier. Note that projects that were created with InstallShield MultiPlatform or InstallShield Universal cannot be upgraded to InstallShield 2026 Express Edition.

Changes to the List of Supported Versions of Windows for Target Systems

Windows XP SP3 and Windows Server 2003 SP2 are now the minimum versions of Windows that are required for target systems that run the installations that are created in InstallShield.

Removal of Support for Digitally Signing with .spc and .pvk Files

InstallShield no longer has support for using .spc and .pvk files to digitally sign files at build time.

If you configured a release or QuickPatch package in InstallShield 2014 Express Edition or earlier to be digitally signed at run time with .spc and .pvk files, and then you try to open that project in InstallShield 2026 Express Edition, InstallShield displays upgrade warning -6048 (for a release) or -6050 (for a QuickPatch project). The warning explains that InstallShield is removing the .pvk file and the associated password from your project during the upgrade.

Before you can successfully build the release or the patch in InstallShield 2026 Express Edition, you will need to remove the .spc reference from the release or QuickPatch project. You can replace it with a .pfx certificate, or with a reference to a certificate in a certificate store. To learn more, see:

- Digital Signing and Security
- Digitally Signing a Release and Its Files at Build Time
- Signing a QuickPatch Package

If you try to build a release or a patch without removing the .spc reference from it, InstallShield displays build error -7347, stating that the .spc file must be removed.

To learn how to convert an .spc file and a .pvk file to a .pfx file, see Digital Signing and Security.

Removal of SignTool.exe and Signcode.exe from InstallShield Installation

`SignTool.exe` and `Signcode.exe` are no longer installed on your machine when you install InstallShield. If you want to digitally sign your files manually, consider using `SignTool.exe`, which is installed with Visual Studio and included in the Microsoft Windows Software Development Kit (SDK).

Trialware Support

InstallShield no longer includes support for creating the Try and Die or the Try and Buy/Product Activation types of trialware. The Trialware view is no longer included in InstallShield.

Upgrading Projects from InstallShield 2013 Express Edition or Earlier

The following information describes possible upgrade issues that may occur when you upgrade projects that were created with InstallShield 2013 Express Edition and earlier to InstallShield 2026 Express Edition. It also alerts you to possible changes in behavior that you may notice between new InstallShield 2026 Express Edition projects and projects that are upgraded from InstallShield 2013 Express Edition or earlier to InstallShield 2026 Express Edition.

General Information about Upgrading Projects that Were Created in Earlier Versions of InstallShield

If you use InstallShield 2026 Express Edition to open a project that was created with an earlier version, InstallShield 2026 Express Edition displays a message box that asks you if you want to convert the project to the new version. If you reply that you do want to convert it, InstallShield creates a backup copy of the project with a file extension such as .774 before converting it. Delete the .774 part from the original project's file name if you want to reopen the project in the earlier version of InstallShield. Note that you cannot open InstallShield 2026 Express Edition projects in earlier versions of InstallShield.

You can upgrade projects that were created with the following versions of InstallShield Express Edition to InstallShield 2026 Express Edition: InstallShield 2013 Express Edition and earlier, InstallShield 12 Express Edition and earlier, and InstallShield Express 5 and earlier. Note that projects that were created with InstallShield MultiPlatform or InstallShield Universal cannot be upgraded to InstallShield 2026 Express Edition.

Upgrading Projects from InstallShield 2012 Spring Express Edition or Earlier

The following information describes possible upgrade issues that may occur when you upgrade projects that were created with InstallShield 2012 Spring Express Edition and earlier to InstallShield 2026 Express Edition. It also alerts you to possible changes in behavior that you may notice between new InstallShield 2026 Express Edition projects and projects that are upgraded from InstallShield 2012 Spring Express Edition or earlier to InstallShield 2026 Express Edition.

General Information about Upgrading Projects that Were Created in Earlier Versions of InstallShield

If you use InstallShield 2026 Express Edition to open a project that was created with an earlier version, InstallShield 2026 Express Edition displays a message box that asks you if you want to convert the project to the new version. If you reply that you do want to convert it, InstallShield creates a backup copy of the project with a file extension such as .773 before converting it. Delete the .773 part from the original project's file name if you want to reopen the project in the earlier version of InstallShield. Note that you cannot open InstallShield 2026 Express Edition projects in earlier versions of InstallShield.

You can upgrade projects that were created with the following versions of InstallShield Express Edition to InstallShield 2026 Express Edition: InstallShield 2012 Spring Express Edition and earlier, InstallShield 12 Express Edition and earlier, and InstallShield Express 5 and earlier. Note that projects that were created with InstallShield MultiPlatform or InstallShield Universal cannot be upgraded to InstallShield 2026 Express Edition.

Changes that Affect All Projects (New and Upgraded Projects)

Change in Requirements for Target Systems

InstallShield no longer supports the creation of installations for Windows 2000 systems.

InstallShield no longer supports the creation of installations for mobile devices. Thus, the Mobile Devices view and the Smart Device project type are no longer included in InstallShield. If you try to upgrade a Smart Device project from InstallShield 2012 Spring Express Edition or earlier to InstallShield 2026 Express Edition, InstallShield 2026 Express Edition displays an error message and fails to open the project. If you upgrade a project from InstallShield 2012 Spring Express Edition or earlier to InstallShield 2026 Express Edition, and if the project targets desktop platforms and contains other mobile device support, InstallShield removes the mobile device support during the upgrade and logs a warning.

Change in Requirements for Systems Running InstallShield

The minimum operating system requirement for running InstallShield is now Windows XP SP3 or Windows Server 2003 SP2. Previously, the minimum operating system requirement was an RTM version of either of these operating systems.

Upgrading Projects from InstallShield 2012 Express Edition or Earlier

The following information describes possible upgrade issues that may occur when you upgrade projects that were created with InstallShield 2012 Express Edition and earlier to InstallShield 2026 Express Edition. It also alerts you to possible changes in behavior that you may notice between new InstallShield 2026 Express Edition projects and projects that are upgraded from InstallShield 2012 Express Edition or earlier to InstallShield 2026 Express Edition.

General Information about Upgrading Projects that Were Created in Earlier Versions of InstallShield

If you use InstallShield 2026 Express Edition to open a project that was created with an earlier version, InstallShield 2026 Express Edition displays a message box that asks you if you want to convert the project to the new version. If you reply that you do want to convert it, InstallShield creates a backup copy of the project with a

file extension such as .772 before converting it. Delete the .772 part from the original project's file name if you want to reopen the project in the earlier version of InstallShield. Note that you cannot open InstallShield 2026 Express Edition projects in earlier versions of InstallShield.

You can upgrade projects that were created with the following versions of InstallShield Express Edition to InstallShield 2026 Express Edition: InstallShield 2012 Express Edition and earlier, InstallShield 12 Express Edition and earlier and InstallShield Express 5 and earlier. Note that projects that were created with InstallShield MultiPlatform or InstallShield Universal cannot be upgraded to InstallShield 2026 Express Edition.

Upgrading Projects from InstallShield 2011 Express Edition or Earlier

The following information describes possible upgrade issues that may occur when you upgrade projects that were created with InstallShield 2011 Express Edition and earlier to InstallShield 2026 Express Edition. It also alerts you to possible changes in behavior that you may notice between new InstallShield 2026 Express Edition projects and projects that are upgraded from InstallShield 2011 Express Edition or earlier to InstallShield 2026 Express Edition.

General Information about Upgrading Projects that Were Created in Earlier Versions of InstallShield

If you use InstallShield 2026 Express Edition to open a project that was created with an earlier version, InstallShield 2026 Express Edition displays a message box that asks you if you want to convert the project to the new version. If you reply that you do want to convert it, InstallShield creates a backup copy of the project with a file extension such as .771 before converting it. Delete the .771 part from the original project's file name if you want to reopen the project in the earlier version of InstallShield. Note that you cannot open InstallShield 2026 Express Edition projects in earlier versions of InstallShield.

You can upgrade projects that were created with the following versions of InstallShield Express Edition to InstallShield 2026 Express Edition: InstallShield 2011 Express Edition and earlier, InstallShield 12 Express Edition and earlier and InstallShield Express 5 and earlier. Note that projects that were created with InstallShield MultiPlatform or InstallShield Universal cannot be upgraded to InstallShield 2026 Express Edition.

Changes that Affect All Projects (New and Upgraded Projects)

This section describes changes that affect both new projects and projects that are upgraded from earlier versions of InstallShield.

Build Warning -7235 for Express Projects

By default, software identification tagging is enabled in all Express projects. This applies to new projects that you create in InstallShield 2026 Express Edition, as well as projects that you have upgraded from earlier versions of InstallShield to InstallShield 2026 Express Edition.

If you build a release in an Express project without entering data in the required identification tag settings (the Unique ID, Tag Creator, and Tag Creator ID settings in the General Information view), and you leave tagging enabled in the project, build warning -7235 occurs. This build warning explains that the software identification tag could not be created and included in the installation because a specific required setting was left blank. To resolve this warning, enter appropriate value in each specific setting, or select No for the Use Software Identification Tag setting in the General Information view.

COM Extraction Changes

InstallShield supports a new monitoring method for COM extraction. If you are using InstallShield on a Windows Vista or later system or a Windows Server 2008 or later system, this new method is used by default. The method uses a kernel driver to monitor the areas of the registry that are modified during dynamic COM extraction at build time and static COM extraction at design time. It combines the advantages that the earlier methods provided, allowing the DLL to read existing registries entries and preventing changes to the build machine.

If necessary, you can switch between the three different COM extraction methods by setting the value data of the UseAPIRegistryHooks registry value, which is in the registry key HKEY_LOCAL_MACHINE\SOFTWARE\InstallShield\RegSpy (on 32-bit machines) or HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\InstallShield\RegSpy (on 64-bit machines). Possible REG_DWORD value data are:

- **0**—Use API hooking to read existing registry entries for the DLL.
- **1**—Use registry redirection to prevent making changes to the registered DLLs on the build machine. If the value is not set, this is the default behavior on Windows XP and Windows Server 2003 systems.
- **2**—Use the new kernel mode monitoring, which combines the advantages of both of the other methods. If the value is not set, this is the default behavior on Windows Vista and later systems and on Windows Server 2008 and later systems.

This functionality applies to Express projects.

Upgrading Projects from InstallShield 2010 Express Edition or Earlier

The following information describes possible upgrade issues that may occur when you upgrade projects that were created with InstallShield 2010 Express Edition and earlier to InstallShield 2026 Express Edition. It also alerts you to possible changes in behavior that you may notice between new InstallShield 2026 Express Edition projects and projects that are upgraded from InstallShield 2010 Express Edition or earlier to InstallShield 2026 Express Edition.

General Information about Upgrading Projects that Were Created in Earlier Versions of InstallShield

If you use InstallShield 2026 Express Edition to open a project that was created with an earlier version, InstallShield 2026 Express Edition displays a message box that asks you if you want to convert the project to the new version. If you reply that you do want to convert it, InstallShield creates a backup copy of the project with a file extension such as .770 before converting it. Delete the .770 part from the original project's file name if you want to reopen the project in the earlier version of InstallShield. Note that you cannot open InstallShield 2026 Express Edition projects in earlier versions of InstallShield.

You can upgrade projects that were created with the following versions of InstallShield Express Edition to InstallShield 2026 Express Edition: InstallShield 2010 Express Edition and earlier, InstallShield 12 Express Edition and earlier and InstallShield Express 5 and earlier. Note that projects that were created with InstallShield MultiPlatform or InstallShield Universal cannot be upgraded to InstallShield 2026 Express Edition.

Changes that Affect All Projects (New and Upgraded Projects)

This section describes changes that affect both new projects and projects that are upgraded from earlier versions of InstallShield.

End of Integration Support for Visual Studio 2003 and Earlier

If you want to create, edit, and build your InstallShield projects directly within Visual Studio, you must use Visual Studio 2005 or later. InstallShield can no longer be integrated with Visual Studio 2003 or earlier.

InstallShield Builds Only Unicode Versions of Setup.exe and Update.exe; Ability to Create ANSI Versions Is No Longer Available

Now all `Setup.exe` and `Update.exe` files that are built in all project types in InstallShield are Unicode. This applies to all new Express and QuickPatch projects that you create in InstallShield 2026 Express Edition. It also applies to all projects that you have upgraded from earlier versions of InstallShield to InstallShield 2026 Express Edition. Therefore, the settings that previously enabled you to specify whether you wanted to build a Unicode version or an ANSI version of the setup launcher have been removed:

- The Setup Launcher Type setting on the Setup.exe tab for a release in the Releases view has been removed from Express projects.
- The Update Launcher Type setting was removed from the Advanced tab in the Build Settings area of the General Information view of QuickPatch projects.

New Name for the Postbuild Tab in the Releases View

The Postbuild tab, which is one of the tabs that InstallShield displays when you select a release in the Releases view, has been renamed. It is now called the Events tab. This change is reported for informational purposes.

Changes that Affect New Projects but Not Upgraded Projects

This section describes changes to InstallShield that may affect new projects but not projects that are upgraded from earlier versions. Note that you may need to make manual changes to upgraded projects.

Changing Design-Time and Build-Time Locations of Existing InstallShield Prerequisites in Existing Projects

InstallShield now lets you specify the folders where InstallShield should search for InstallShield prerequisite files (.prq files), their associated data files, and their dependencies. Previously, InstallShield searched for .prq files in the following location only: *`InstallShield Program Files Folder`*`\SetupPrerequisites`.

If you move any InstallShield prerequisites from the *`InstallShield Program Files Folder`*`\SetupPrerequisites` folder to a new custom location that you have defined on the Prerequisites tab of the Options dialog box (or any of the other places where search paths can be defined now), you may need to perform the following steps in InstallShield 2010 Express Edition or earlier projects when you upgrade them to InstallShield 2026 Express Edition:

1. In the Redistributables view, clear the check box for each InstallShield prerequisite that is included in your project but is located in a custom location. Also clear the check box for each InstallShield prerequisite whose data files or dependencies were moved from the default location to a custom location.
2. Click the new Refresh button.
3. Select the check box for each InstallShield prerequisite that you removed from your project in step 1.

InstallShield removes the path of the prerequisite from the `ISSetupPrerequisites` table of your project. The full path was stored in this table in InstallShield 2010 Express Edition and earlier projects. Note that if you only clear a prerequisite's check box and then reselect it without clicking the Refresh button, InstallShield continues to use the full path, rather than just the file name, in the `ISSetupPrerequisites` table.

If you upgrade an InstallShield 2010 Express Edition or earlier project to InstallShield 2026 Express Edition, change the location of an InstallShield prerequisite, and then add that prerequisite to your project, you do not need to perform the refresh procedure. Also, if you create a new project in InstallShield 2026 Express Edition, you do not need to perform the refresh procedure. In both cases, InstallShield does not include the path in the `ISSetupPrerequisites` table of your project, which enables you to use the custom search path, instead of the default path.

Upgrading Projects from InstallShield 2009 Express Edition or Earlier

The following information describes possible upgrade issues that may occur when you upgrade projects that were created with InstallShield 2009 Express Edition and earlier to InstallShield 2026 Express Edition. It also alerts you to possible changes in behavior that you may notice between new InstallShield 2026 Express Edition projects and projects that are upgraded from InstallShield 2009 Express Edition or earlier to InstallShield 2026 Express Edition.

General Information about Upgrading Projects that Were Created in Earlier Versions of InstallShield Express Edition

If you use InstallShield 2026 Express Edition to open a project that was created with an earlier version, InstallShield 2026 Express Edition displays a message box that asks you if you want to convert the project to the new version. If you reply that you do want to convert it, InstallShield creates a backup copy of the project with a file extension such as .768 before converting it. Delete the .768 part from the original project's file name if you want to reopen the project in the earlier version of InstallShield. Note that you cannot open InstallShield 2026 Express Edition projects in earlier versions of InstallShield Express Edition.

You can upgrade projects that were created with the following versions of InstallShield Express Edition to InstallShield 2026 Express Edition: InstallShield 2009 Express Edition and earlier, InstallShield 12 Express Edition and earlier and InstallShield Express 5 and earlier. Note that projects that were created with InstallShield MultiPlatform or InstallShield Universal cannot be upgraded to InstallShield 2026 Express Edition.

Installing More than One Edition of InstallShield

Only one edition of InstallShield 2010— Premier, Professional, or Express—can be installed on a system at a time. Previously, it was possible to install the Express edition on the same system that had the Premier or Professional edition of the same InstallShield version.

Change to the List of Supported Operating Systems for Running InstallShield

The minimum operating system requirement for systems that run InstallShield (the authoring environment) is now Windows XP or Windows Server 2003. Previously, the minimum operating system requirement was Windows 2000 SP3.

Changes that Affect All Projects (New and Upgraded Projects)

This section describes changes that affect both new projects and projects that are upgraded from earlier versions of InstallShield.

Setup.exe No Longer Runs on Windows 9x, Windows NT 4, or Windows Me Systems

`Setup.exe` installations that are created in InstallShield can no longer be run on Windows 9x, Windows NT 4, or Windows Me. If an end user tries to launch `Setup.exe` on a Windows 9x or Windows Me system, Windows displays a message box with the following error: "The *`FullSetup.exePathAndFileName`* file expects a newer version of Windows. Upgrade your Windows version." On Windows NT 4 systems, Windows displays a message box with the following error: "*`FullSetup.exePathAndFileName`* is not a valid Windows NT application."

InstallShield no longer enables you to select these legacy operating systems when you are creating conditions for a feature or a custom action, or when you are configuring system hardware requirements in the Requirements view. If you upgrade a project that was created in InstallShield 2009 Express Edition or earlier to InstallShield 2026 Express Edition, and if the earlier project had conditions or requirements for only these legacy operating systems, InstallShield replaces the legacy operating system options with the Any OS Version option.

Windows Installer 1.x Redistributables Are No Longer Available

InstallShield no longer includes Windows Installer 1.x redistributables, since they target only legacy versions of Windows that are no longer supported. Previously, it was possible to add Windows Installer 1.x redistributables to a project through the Releases view.

Redistributable for VBScript Runtime Files Is No Longer Available

InstallShield no longer includes the InstallShield object for VBScript Runtime Files. This redistributable targets only legacy versions of Windows that are no longer supported.

QuickPatch Creation

InstallShield now uses the Windows Installer 4.5 patching technology to create QuickPatch releases. This change is reported for informational purposes.

Changes for the Redistributables View

The Redistributables view has a new toolbar and group box area that provide robust search and organizational functionality. Use the new Show Details button in this view to show or hide the details pane for the selected redistributable in this view. The details pane provides information such as which files a redistributable installs. The Show Details button replaces the Show Details and Hide Details links that were previously available in the upper-right corner of this view.

The new group box area is below the new toolbar in the Redistributables view. You can drag and drop column headings onto this group box area to organize the list of redistributables in a hierarchical format. If you want InstallShield to separate all of the redistributables in the view into two groups—one whose check box is selected and one whose check box is cleared—drag the check box column to the group box area. This enables you to easily identify all of the redistributables that are included in your project. The result is similar to the behavior that previously occurred if you right-clicked any redistributable and then clicked Show Only Selected Items. Note that the Show Only Selected Items command is no longer available in the Redistributables view.

For more information, see Working with the Group Box Area in Various Views.

Changes that Affect New Projects but Not Upgraded Projects

This section describes changes to InstallShield that may affect new projects but not projects that are upgraded from earlier versions. Note that you may need to make manual changes to upgraded projects.

Changes to Support for Securing Permissions for Files, Folders, and Registry Keys

The General Information view has a new Locked-Down Permissions setting that lets you specify whether you want to use the new custom InstallShield handling or the traditional Windows Installer handling for all new permissions that you set for files, folders, and registry keys in your project. The new custom InstallShield handling option offers several advantages over the traditional Windows Installer handling option.

In all new projects, the default value for this setting is the custom InstallShield handling option. If you upgrade a project from InstallShield 2009 Express Edition or earlier to InstallShield 2026 Express Edition, the traditional Windows Installer handling option is the default value of this setting.

This new setting is available in Express projects.

For more information, see the following:

- Securing Files, Folders, and Registry Keys in a Locked-Down Environment
- Selecting the Locked-Down Permissions Type for a Project

Changes to the Ready to Install Dialog for Beta Windows Installer 5 Support of Per-User Installations

The Show All Users Option setting in the Dialogs view now has support for the MSIINSTALLPERUSER property that is available with the beta of Windows Installer 5. Use this setting to specify whether you want to give end users the option of installing your product for all users or for only the current user. Depending on the value that you select for this setting, the Ready to Install dialog may include buttons that let end users specify how they want to install the product; the buttons are displayed if the installation is run on a system that has Windows 7 or Windows Server 2008 R2.

The Show All Users Option setting is now available when you select the main Dialogs node in the Dialogs view. Previously, this setting was available if you selected the Customer Information dialog in this view.

If you create a new Express project in InstallShield 2026 Express Edition, the default value for the Show All Users Option setting is No. If you upgrade a project from InstallShield 2009 Express Edition or earlier to InstallShield 2026 Express Edition, the value is set as follows:

- If No was selected in the earlier project, No is selected in the upgraded project.
- If Yes was selected in the earlier project, Yes (All Systems) is selected in the upgraded project. Therefore, if the target system has Windows 7 or Windows Server 2008 R2, the Ready to Install dialog includes buttons that let end users specify how they want to install the product. If the target system has Windows Vista or earlier, or Windows Server 2008 or earlier, the Customer Information dialog includes radio buttons let end users specify how they want to install the product.

You can modify the value as needed.

Trialware Support

The only edition of InstallShield that includes the Trialware view is the Premier edition. This edition lets you create the Try and Die type of trialware. InstallShield no longer includes support for creating the Try and Buy/Product Activation type of trialware.

Compact Projects

InstallShield no longer has support for Compact projects.

Visual Studio Integration

Microsoft Visual Studio can be integrated with only one version of InstallShield Express Edition at a time. The last version of InstallShield that is installed or repaired on a system is the one that is used for Visual Studio integration.

Upgrading Projects from InstallShield 2008 Express Edition or Earlier

The following information describes changes that may affect projects that are upgraded from InstallShield 2008 Express Edition or earlier to InstallShield 2026 Express Edition.

Upgrading Projects Created in Earlier Versions of InstallShield Express Edition

If you use InstallShield 2026 Express Edition to open a project that was created with an earlier version, InstallShield 2026 Express Edition displays a message box that asks you if you want to convert the project to the new version. If you reply that you do want to convert it, InstallShield creates a backup copy of the project with a file extension such as .765 before converting it. Delete the .766 part from the original project's file name if you want to reopen the project in the earlier version of InstallShield. Note that you cannot open InstallShield 2026 Express Edition projects in earlier versions of InstallShield Express Edition.

You can migrate projects that were created with the following versions of InstallShield Express Edition to InstallShield 2026 Express Edition: InstallShield 12 Express Edition and earlier and InstallShield Express 5 and earlier. Note that projects that were created with InstallShield MultiPlatform or InstallShield Universal cannot be migrated to InstallShield 2026 Express Edition.

Changes that Affect All Projects (New and Upgraded Projects)

This section describes changes that affect both new projects and projects that are upgraded from earlier versions of InstallShield.

New Default Setup Launcher Value for New Releases and New QuickPatch Projects: Windows Installer Is Not Included

When you create a new Express or QuickPatch project, the redistributable for the Windows Installer engine is no longer included by default:

- In Express projects, the default value for the Setup Launcher setting on the Setup.exe tab is now **Yes (no MSI engine included)**. Previously, the default value for this setting was **Yes (include Windows NT & Windows 9x MSI engine)**.
- In QuickPatch projects, the **Include Windows Installer 2.0 engine** and **Include Windows Installer 3.0 engine** check boxes on the Common tab in the General Information view are cleared by default. Previously, these check boxes were selected by default.

These changes apply to all new Express and QuickPatch projects that are created in InstallShield 2026 Express Edition.

If you upgrade an Express or QuickPatch project from InstallShield 2008 Express Edition or earlier to InstallShield 2026 Express Edition, InstallShield does not automatically change the values of the aforementioned settings.

File Compression for Files that Are Streamed into Setup.exe and ISSetup.dll at Build Time

If you build a release that uses a `Setup.exe` setup launcher, InstallShield now compresses files that it streams into the `Setup.exe` file at build time. The default compression level that InstallShield uses offers a balance between file size and time that is required to extract the compressed files at run time. This applies to new Express projects as well as existing Express projects that are upgraded from InstallShield 2008 Express Edition or earlier to InstallShield 2026 Express Edition.

If you want to change the compression level or you do not want to use any compression, you can override the default level through a machine-wide setting. For more information, see Configuring the Compression Level for Files that Are Streamed into Setup.exe.

Previously, InstallShield did not include any support for compressing files that were streamed into the `Setup.exe` file at build time. Thus, if you compare a release that was built in InstallShield 2008 Express Edition or earlier with the same release that is built with the default compression level in InstallShield 2026 Express Edition, you may notice that the file size of `Setup.exe` is slightly different. In addition, the time that is required to extract files may be slightly different.

Multi-Part .cab Files

InstallShield now has a default limit of 600 MB for each .cab file that it creates at build time for a SingleImage release where all of the files are embedded in a single-file .msi package or a `Setup.exe` setup launcher. When InstallShield is creating the .cab files for this type of release and it reaches this limit, it splits the data into two or more .cab files, creating multi-part .cab files. This applies to new Express projects as well as existing Express projects that are upgraded from InstallShield 2008 Express Edition or earlier to InstallShield 2026 Express Edition.

You can modify the .cab size limit if necessary. In addition, if you do not want InstallShield to create multi-part .cab files, you can configure it to create single .cab files. For more information, see Configuring the Maximum Size for .cab Files.

Previously, InstallShield did not create multi-part .cab files, and there was no built-in limit for the .cab file size.

Proxy Server Support

You may want to configure your installation to download certain files only if they are needed on the target system. For example, the Windows Installer engine, the .NET Framework, and some InstallShield prerequisites may already be present on some or most target systems. Instead of embedding these files in your installation (which would increase your overall installation size), you can configure your project so that only the ones that are needed are downloaded at run time.

If your end users access the Internet through a proxy server and your installation is configured to download files, the installation now uses the system proxy settings that are manually configured in Internet Explorer during the download. This occurs even if another browser on the target system is the default browser.

Note that InstallShield does not include support for the Automatically Detect Settings functionality in Internet Explorer. (If end users have the Automatically Detect Settings check box selected in Internet Explorer for their LAN connection and the installation needs to download files, the installation fails because the files cannot be downloaded. If it is possible that your end users may have the Automatically Detect Settings check box selected in Internet Explorer for their LAN connections, you may want to embed all of the files in your installation rather

than configure them to be downloaded; if the files are embedded, the failures can be avoided.) However, InstallShield does support the Automatic Configuration Script functionality that is set up for LAN connections in Internet Explorer.

This is the behavior for all new projects in InstallShield 2009, as well as all projects that are created in earlier versions and then upgraded to InstallShield 2009.

In InstallShield 2008 and earlier, the installation attempted to use the proxy server settings that were configured in whatever browser was the default browser. However, this was not always possible, and it caused some problems:

- If Netscape 6 or 7 was the default browser, the Netscape 4 settings were used. If Netscape 8 or 9 was the default browser, the system (Internet Explorer) settings were used.
- If Netscape 4 settings were used, only the proxy server list was read and imported correctly. The proxy bypass list was read, but it was not imported correctly.
- Non–Internet Explorer 4 compatible settings such as the auto-proxy script setting were not imported.
- The method that the installation used for determining the default browser was not compatible on Windows Vista. Therefore, on Windows Vista systems, an installation may not have detected the default browser correctly.

Changes that Affect New Projects but Not Upgraded Projects

This section describes changes to InstallShield that may affect new projects but not projects that are upgraded from earlier versions. Note that you may need to make manual changes to upgraded projects.

Ability to Create Unicode Versions of the Setup.exe and Update.exe Bootstrappers

InstallShield now enables you to specify whether you want to create a Unicode version or an ANSI version of the `Setup.exe` setup launcher for a project. Previously, if your project included a setup launcher, InstallShield always built an ANSI version; it did not include support for building a Unicode version.

A Unicode setup launcher can correctly display double-byte characters in the user interface of the setup launcher, regardless of whether the target system is running the appropriate code page for the double-byte-character language. An ANSI setup launcher displays double-byte characters in the setup launcher dialogs if the target system is running the appropriate code page. However, it displays garbled characters instead of double-byte characters in those dialogs if the target system is not running the appropriate code page.

If you create a new Express project in InstallShield 2026 Express Edition, the default setup launcher type is Unicode. In addition, if you create a new QuickPatch project in InstallShield 2026 Express Edition, the default update launcher type is Unicode.

If you upgrade an Express project or a QuickPatch project from InstallShield 2008 Express Edition or earlier to InstallShield 2026 Express Edition, the update launcher type for any existing patch is ANSI. You can override the type if appropriate.

Dynamic File Links

When you add or modify a dynamic file link in a project, you can specify which component creation method you want InstallShield to use: a new best practice method or the previously available one-component-per-directory method. These methods are applicable to dynamic file linking in Express projects.

If you create a new dynamic file link in InstallShield 2026 Express Edition, InstallShield uses the best practice method by default.

All dynamic file links that are created in InstallShield 2008 Express Edition or earlier use the one-component-per-directory method. If you have a project with dynamic file links and you upgrade it from InstallShield 2008 Express Edition or earlier to InstallShield 2026 Express Edition, InstallShield continues to use the one-component-per-directory method for creating the components of those already present dynamic file links. For any new dynamic file links that you create in the upgraded project, the best practice method is used by default. For detailed information about the two component creation methods, as well as guidance on which method you should use, see Determining the Appropriate Component Creation Method for Dynamically Linked Files.

IIS Web Sites Without Virtual Directories

InstallShield now includes support for installing IIS Web sites without any virtual directories. This support is available for all new Web sites that are created in new InstallShield 2026 Express Edition projects. This support is also available if you upgrade a project from InstallShield 2008 Express Edition or earlier to InstallShield 2026 Express Edition and then add a new Web site.

Note that if you upgrade an InstallShield 2008 or earlier project to InstallShield 2026 Express Edition, and the project already contains a Web site, the Web site cannot be installed if it does not have any virtual directories. In order to be able to install the Web site without virtual directories, you must manually delete it from your InstallShield 2026 Express Edition project, and then re-add it to your project as a new Web site.

Simplification of QuickPatch Packages

The new Streamline QuickPatch setting on the Advanced tab in a QuickPatch project determines how InstallShield builds QuickPatch packages. A streamlined QuickPatch package typically has fewer new subfeatures and custom actions than a non-streamlined QuickPatch package.

In some cases, InstallShield cannot streamline the QuickPatch package. For example, if you configure the QuickPatch package to remove an installed file, InstallShield cannot streamline it.

When you create a new QuickPatch project, the default value for the Streamline QuickPatch setting is Yes. However, when you upgrade a QuickPatch project from InstallShield 2008 Express Edition or earlier to InstallShield 2026 Express Edition, the value for this setting is No. You can change this value if appropriate. For more information, see Specifying Whether to Streamline the QuickPatch Package.

Default Condition for the SetARPINSTALLLOCATION Custom Action

By default, all new Express projects contain the built-in InstallShield custom action SetARPINSTALLLOCATION. This custom action, which sets the value of the `ARPINSTALLLOCATION` property to the fully qualified path for the product's primary folder, is scheduled for the Installation Execute sequence, and it has no condition. In InstallShield 2008 Express Edition and earlier, the default condition for this custom action was Not Installed. With this default Not Installed condition, the custom action is not run during maintenance mode, and this results in a blank value for the `ARPINSTALLLOCATION` property.

Upgrading Projects from InstallShield 12 Express Edition or Earlier

The following information describes changes that may affect projects that are upgraded from InstallShield 12 Express Edition or earlier to InstallShield 2026 Express Edition.

Upgrading Projects Created in Earlier Versions of InstallShield Express Edition

If you use InstallShield 2026 Express Edition to open a project that was created with an earlier version, InstallShield 2026 Express Edition displays a message box that asks you if you want to convert the project to the new version. If you reply that you do want to convert it, InstallShield creates a backup copy of the project with a file extension such as .765 before converting it. Delete the .765 part from the original project's file name if you want to reopen the project in the earlier version of InstallShield. Note that you cannot open InstallShield 2026 Express Edition projects in earlier versions of InstallShield Express Edition.

You can migrate projects that were created with the following versions of InstallShield Express Edition to InstallShield 2026 Express Edition: InstallShield 12 Express Edition and earlier and InstallShield Express 5 and earlier. Note that projects that were created with InstallShield MultiPlatform or InstallShield Universal cannot be migrated to InstallShield 2026 Express Edition.

End of Support for Windows 9x, Windows NT 4, and Windows Me on Target Systems

InstallShield no longer supports the creation of installations for Windows 9x, Windows NT 4, and Windows Me systems. If end users have one of these operating systems on their computer and they try to run an installation that was built with InstallShield 2026 Express Edition, unexpected results may occur.

COM Extraction

Unused Directories Automatically Removed from the .msi File at Build Time by Default

Note that if you upgrade an Express project that was created in InstallShield 12 Express Edition or earlier to InstallShield 2026 Express Edition, the new Keep Unused Directories setting on the Build tab in the Releases view is set to No. Therefore, if a directory that is listed in the Directory column of the `Directory` table is not referenced in any known location in the .msi file, InstallShield removes it from the `Directory` table of the .msi file that it creates at build time. This occurs after any merge modules are merged, but only directories that are present in the .msi file are removed; therefore, if a merge module contains new unused directories in its `Directory` table, the new unused directories are added to the installation.

In some cases, you may want to change the value of the Keep Unused Directories setting to Yes. For example, if your project includes custom actions that use directories that are not referenced in any other area of the project, you may want to select Yes for the Keep Unused Directories setting.

Automatic Downgrade Prevention

When you create a new Express project, the Upgrade Paths view contains an upgrade item called ISPreventDowngrade; ISPreventDowngrade prevents the current installation from overwriting a future major version of your product. If you migrate a project from InstallShield 12 Express Edition or earlier to InstallShield 2026 Express Edition, the ISPreventDowngrade item is not automatically added.

For instructions on how to manually add downgrade prevention support, see Preventing the Current Installation from Overwriting a Future Major Version of the Same Product.

Support for the UAC Shield Icon on Dialog Buttons

The Install button on the Ready to Install dialog and the Remove button on the Ready to Remove dialog have the User Account Control (UAC) shield icon when the installation is run on Windows Vista systems and the installation is not yet running with elevated privileges. This applies to all new projects as well as projects that are migrated from InstallShield 12 Express Edition or earlier.

Note that InstallShield is run with elevated privileges. Therefore, if you launch your installation from within InstallShield on a Windows Vista system, it has elevated privileges, and the UAC shield icon is not displayed on the Ready to Install and Ready to Remove dialogs.

Changes for ALLUSERS and for the Customer Information Dialog

Beginning with InstallShield 2008 Express Edition, the ALLUSERS property is set to 1 by default in all new Express projects. This is the recommended implementation, since most installations must be run in a per-machine context with administrative privileges.

If you upgrade a project that was created with InstallShield 12 Express Edition or earlier to InstallShield 2026 Express Edition, InstallShield does not automatically change the value of the ALLUSERS property or add this property if it was not defined in the earlier project.

If you want to change the value of ALLUSERS in a new or migrated project, you can do so through the General Information view. For more information, see General Information Settings.

Also new with InstallShield 2008 Express Edition, by default, the Customer Information dialog in all new Express projects does not display the radio button group that enables end users to specify whether they want to install the product for all users or for only the current user. This is the recommended implementation for this dialog.

If you upgrade a project that was created with InstallShield 12 Express Edition or earlier to InstallShield 2026 Express Edition, InstallShield does not automatically change the Customer Information dialog.

To learn more, see Per-User vs. Per-Machine Installations.

New Default Value for the Cache Path Setting for a Release

The default value for the Cache Path setting for a compressed release in the Releases view is now set to *[LocalAppDataFolder]Downloaded Installations*. The previous default value was *[WindowsFolder]Downloaded Installations*, which may not be available to users on locked-down systems. If you migrate a project from InstallShield 12 Express Edition or earlier to InstallShield 2026 Express Edition, the Cache Path setting is not automatically changed. Therefore, you may want to change that value.

Patch Creation for QuickPatch Projects

InstallShield now uses version 3.1 of Patchwiz.dll to create QuickPatch packages.

DemoShield Support

DemoShield is no longer being sold. In addition, it is no longer supported. Therefore, InstallShield no longer includes any DemoShield integration.

Upgrading Projects from InstallShield Express 2.x

There are many technical differences between the current version of InstallShield and those prior to InstallShield Express 3.0. InstallShield Express 2.x is based upon traditional setup architecture, where an executable file called Setup.exe is created. This executable contains all the logic necessary to perform your installation. In the current version of InstallShield, the output file is or contains a relational database (.msi) containing all the needed information to interact with the Windows Installer service. Due to these vast differences, there may be aspects of

your earlier InstallShield project that do not fit into the current architecture. Additionally, there will be gaps in the new project created as a result of the migration. These missing pieces must be filled in before you can build and distribute your setup.

Project Types

In previous versions of InstallShield, when creating a new setup project you were prompted to select a project type. The current version of InstallShield has no equivalent for project types. The type of project you create depends upon the files you add to your setup and the system changes you make. All the information gathered from selecting a certain project type is migrated to your new InstallShield project.

Files, File Groups, and Components

All files you have added to your setup are transferred into your new InstallShield project. In older versions of InstallShield, files were added to file groups and file groups were added to components. In the current version of InstallShield, files are added to features. No further levels exist. Therefore, when you migrate your setup project you no longer have files associated with file groups. Instead, all files belong to a feature. For every component you have in your old project, a feature is created in your new project. Your file groups map to destination folders. For more details, see Destination Folders.

Setup Types

Setup types provide similar functionality in the current version of InstallShield as they did in earlier versions. The major difference between the two is the fact that setup types used to be based upon components. In the new version of InstallShield, components do not exist. Instead they have been replaced by features. The setup types you specified in your old InstallShield Express project are re-created in the new InstallShield. To learn more, see Working with Setup Types.

InstallShield Objects

InstallShield objects have been replaced with merge modules. If a merge module that can supply the functionality provided by the object you used in your project exists, that module is added to your setup. If your setup includes a file that is also included in a merge module, the module is added to your setup rather than your single file.

To learn more, see Including Redistributables in Your Installation.

InstallShield Extensions

InstallShield extensions have been replaced by custom actions. With custom actions you can call a function from a DLL or launch an executable. If you had an extension that performed one of these two tasks, that extension is converted into a custom action in your new project. For more information, see Using Custom Actions.

Registry Entries

All of the registry data in your old project is added to your new project.

Shortcuts

Shortcuts in the new InstallShield are much more robust than those found in previous versions. Any shortcuts you had in your old setup are migrated into your new setup project. However, since there are many more settings for shortcuts in the latest version of InstallShield, you may want to go to the Shortcuts/Folders view and configure the settings for each of the shortcuts your project contains, such as the hot key combination, the icon index, and the working directory. These settings are not required, however.

For specific details about each of the settings, see Shortcut Settings.

Dialogs

Although the run-time dialogs in the latest version of InstallShield differ in look from those in earlier versions, they serve similar functions. The Setup Type dialog, for example, still serves the same purpose it always has: it allows your customers to select which setup type they want to use. The same dialogs you chose to include in your old InstallShield Express project are included in your new project.

One major difference between the dialogs in earlier versions of InstallShield Express and those in the latest version is the use of .rtf files. When linking to a license agreement or readme file in earlier versions of InstallShield Express, you would use a text (.txt) file. The latest version of InstallShield requires you to use rich text (.rtf) files. Your previously created .txt files are converted to .rtf files as part of the migration process.

Distribution Media

The type of media you selected in previous versions of InstallShield Express is not migrated into your new InstallShield setup. Select a medium from the list of available media types in the Build Your Release view. For information about each of the available types, see Media Types.

If you had previously released your setup on floppy disk, see Building a Release for Floppy Disk Distribution.

16-bit Setups

Because the Windows Installer service runs only on 32-bit operating systems, your 16-bit setups do not migrate as well as 32-bit setups. All of the files, shortcuts, and other settings you created for your 16-bit setup are migrated, but any self-registering 16-bit files you include in your setup do not self-register. In reality, since your setup cannot run on a 16-bit system, there is little reason for you to have 16-bit files included in the setup.

Languages

InstallShield has built-in support for many languages. If the language of your old InstallShield Express project matches one of the supported languages, your new InstallShield setup is created with the run-time strings for that language. For example, if you have a setup that is localized to French and migrate that setup to the latest version of InstallShield, all the strings in the new InstallShield project are in French. If your old InstallShield Express project runs in an unsupported language, the run-time strings for your new project appear in English. This does not mean, however, that your setup cannot run in your desired language. You can translate all the run-time strings in the Text and Messages view.

Upgrading to the Premier or Professional Editions of InstallShield

If you find that your installation needs outgrow the functionality that is available in the Express edition of InstallShield, you can easily take your existing project and use it to create an installation with the Premier or Professional editions of InstallShield. The Premier and Professional editions have greater customization features that more complex installations require.

Features that Are in Only the Premier Edition

Following is a list of some of the features that are available in the Premier edition but not the Professional or Express editions:

- **Ability to create and build Suite/Advanced UI installations**—Create a bootstrap application with a contemporary, customizable user interface for multiple .msi packages, .msp packages, InstallScript packages, .exe packages, sideloading app packages (.appx), and Windows Installer transactions, as well as multiple InstallShield prerequisites. A Suite/Advanced UI installation packages together multiple separate installations as a single installation while providing a unified user interface; it uses a setup launcher (`Setup.exe`) to conditionally launch packages on target systems as needed.
- **Support for InstallScript actions and other types of actions in Suite/Advanced UI installations**—Suite/Advanced UI installations have built-in support for launching different types of actions to perform various run-time tasks that are outside the scope of the packages that you are including in the installation. The actions can run executable files, call DLL functions, run PowerShell scripts, set a Suite/Advanced UI property, run InstallScript code, or call a public method in a managed assembly at run time.
- **Virtualization support**—The Microsoft App-V Assistant is included in the Virtualization Pack. Use this assistant to create customized virtual applications in the Microsoft App-V format. Virtualization enables you to isolate an application in its own environment so that it does not conflict with existing applications or modify the underlying operating system.
- **Application Virtualization Suitability Suites**—Several validation suites are available in InstallShield for helping you to determine how ready your products are for virtualization. The InstallShield virtualization internal consistency evaluators (ISVICEs) that are included in these suites let you check suitability for Microsoft App-V 4.x, Microsoft App-V 5.x, Microsoft Server App-V, VMware ThinApp, and Citrix XenApp. The validation suites can enable you to make more informed decisions about how you should build your product if you are considering offering your customers a virtualized version.
- **Support for DIM files**—The ability to create DIM projects is available in the Premier edition of InstallShield. This support is also available in the InstallShield Developer Installation Manifest Editor, a collaboration add-on. The ability to add DIM files to Basic MSI projects is available in the Premier edition of InstallShield.

  A DIM project is a feature-sized collection of related items such as product files, shortcuts, registry entries, text file changes, IIS Web sites, and other elements that together make up a discrete portion of a product installation. Working with DIMs enables multiple team members to contribute to the development of the installation simultaneously. Each software developer or other team member can work on a separate DIM that the release engineer can reference in one or more Basic MSI projects.
- **Multilingual installations**—Create a single installation that displays end-user text in multiple languages and can handle conditional installation of language-specific files. Change dialogs and messages to any one of 34 additional languages using pre-translated strings.

Project ▪ *Note that support for two of those languages—Arabic (Saudi Arabia) and Hebrew—is available in only Basic MSI and Merge Module projects.*

- **Arabic (Saudi Arabia) and Hebrew language support**—InstallShield Premier Edition includes support for Arabic (Saudi Arabia) and Hebrew languages, which are written and read from right to left. All of the default end-user dialog strings are available in these languages.

  Since these languages are read from right to left, the Premier edition also includes support for mirroring Arabic and Hebrew dialogs; that is, InstallShield uses a right-to-left layout for Arabic and Hebrew dialogs. Thus, for example, buttons that are on the right side of dialogs in English and other left-to-right languages are moved to the left side of right-to-left-language dialogs.

- **Ability to specify commands that run before, during, and after builds**—InstallShield Premier Edition includes release settings that you can use to specify commands that you want to be run at various stages of the build process. You can schedule commands that run at the following build events: (a) before InstallShield starts building the release, (b) after InstallShield has built the .msi package and the .cab files (if your product's data files are to be stored in .cab files), but before the .msi package has been digitally signed and streamed into the `Setup.exe` file, and (c) after InstallShield has built and signed the release.

- **Ability to distribute installations to virtual machines that InstallShield provisions at build time or on demand**—You can configure your projects so that after each successful build of your installation, InstallShield automatically reverts a virtual machine (VM) to a designated snapshot, powers on the VM, and copies your installation to the VM to make it available for testing. You can also alternately perform these testing preparation steps on demand at any time. The testing preparation capability makes it possible to reduce testing time and eliminate manual steps. The VM can be on a Microsoft Hyper-V Server, a VMware ESX or ESXi Server, a VMware Workstation, a Microsoft Virtual PC, a Xen, an Amazon EC2 Instance, a QEMU, a Parallels Desktop, a Google Compute, a Microsoft Azure, or an EverRun.

- **Extra licenses for InstallShield MSI tools**—InstallShield includes several tools: InstallShield MSI Diff, InstallShield MSI Query, InstallShield MSI Sleuth, and InstallShield MSI Grep. You can use these tools to troubleshoot issues with Windows Installer packages. InstallShield Premier Edition includes a separate installation and extra licenses that let you install just the InstallShield MSI tools, without InstallShield, on separate machines. For specific terms, see the End-User License Agreement for the InstallShield MSI tools.

- **Ability to import IIS data from existing IIS Web sites into a project**—InstallShield includes an IIS scanner (`IISscan.exe`), a command-line tool for scanning an existing IIS Web site and recording IIS data about the Web site. The IIS scanner creates an XML file that contains all of the settings for the Web site, its virtual directories, its applications, and its application pools. You can use the XML file to import the IIS data into the Internet Information Services view in InstallShield Premier Edition. Once you have imported the IIS data into a project, you can use the Internet Information Services view to make changes to the IIS settings as needed.

- **Support for deploying Web Deploy packages**—Suite/Advanced UI installations, available in the Premier edition of InstallShield, have built-in support for deploying Web Deploy packages to IIS Web servers and the cloud.

- **InstallShield Collaboration**—InstallShield Premier Edition includes licenses for InstallShield Collaboration for Visual Studio.

- **Network repository**—A network repository is a collection of installation elements that multiple installation authors can access and reuse in their projects as needed. A network repository fosters collaboration among installation authors; it is stored on a network.

- **InstallShield Best Practice Suite**—InstallShield includes a set of validators called the InstallShield Best Practice Suite. The InstallShield Best Practice (ISBP) validators in this suite alert you if your installation violates best-practice guidelines.
- **Additional Dialog Themes**—Several dialog themes are available only in the Premier edition of InstallShield.

Features that Are in Only the Premier and Professional Editions

Following is a list of some of the features that are available in the Premier and Professional editions but not the Express edition:

- **Ability to create and build Advanced UI installations**—Create a bootstrap application with a contemporary, customizable user interface for a single .msi package, .msp package, or InstallScript package, as well as multiple InstallShield prerequisites. An Advanced UI installation uses a setup launcher (`Setup.exe`) to conditionally launch packages on target systems as needed.

  (Note that the Suite/Advanced UI functionality in the Premier edition of InstallShield includes extended support for this functionality: The Suite/Advanced UI functionality enables you to package multiple .msi packages, .msp packages, InstallScript packages, .exe packages, sideloading app packages (.appx), and Windows Installer transactions, as well as multiple InstallShield prerequisites into a single installation with a contemporary, unified, customizable user interface.)
- **Standalone Build**—This tool, which is available with the Premier and Professional editions of InstallShield, enables you to install only the part of InstallShield that builds the installations, plus any redistributables that you want to include, on a build machine. Extra licenses of the Standalone Build are available for purchase.
- **Dialog Editor**—The Dialog Editor enables you to modify the layout of existing end-user dialogs or create new custom dialogs. Import and export dialogs to share them across projects. Construct different dialogs for each language supported in the project.
- **InstallShield MSI tools**—The Premier and Professional editions of InstallShield include several tools: InstallShield MSI Diff, InstallShield MSI Query, InstallShield MSI Sleuth, and InstallShield MSI Grep. You can use these tools to troubleshoot issues with Windows Installer packages.
- **Automation interface**—Use script to add new files, add or delete features, change the product name and upgrade code, change release settings, change summary information stream items, change release flags, change any property, initiate the build process, and more.
- **Release customization**—Define which project segments to compress, which features to place on which disk, and which languages to include. Choose to filter application data based on language to support localization efforts.
- **Source code control integration**—Simplify the process of checking projects in and out of your source code control system and save space when differencing projects. The SCC integration in the Premier and Professional editions of InstallShield supports integration with various source code control systems.
- **Flexible localization support**—The Premier and Professional editions of InstallShield include a String Editor view, which gives you complete and centralized control over the localizable text strings that are displayed at run time during the installation process. You can use this view to edit the strings for everything from button text to feature descriptions. This view also includes support for exporting string entries (which you can have translated) and for importing translated string entries back into your project.
- **Project validation**—Use standard .cub files to validate installations and merge modules. Use the upgrade and patching validation to find out about potential upgrade problems and resolve them before you release your upgrades and patches.

- **Patch creation**—In addition to enabling you to create QuickPatch projects, the Premier and Professional editions let you create standard patches that contain updates to a previous version of your product.
- **Manage multiple product versions**—Build versions such as Evaluation, Debug, Standard, and Advanced—from a single project. Allow specific features, InstallShield prerequisites, and other elements to be chosen for inclusion in (or exclusion from) a release through user-defined flags.
- **Support for InstallScript**—The Premier and Professional editions of InstallShield include support for InstallScript, a simple but powerful programming language. You can add InstallScript custom actions to Windows Installer–based installations or create InstallScript projects, which use the InstallScript engine instead of the Windows Installer engine to control the entire installation.
- **Flexible custom action support**—The Premier and Professional editions of InstallShield include support for several custom action types that are not available in the Express edition. These extra custom action types enable you to do the following: set a property, set a directory, call a public method in a managed assembly, or display error message under certain conditions and abort the installation.
- **Flexible shortcut support**—The Premier and Professional editions of InstallShield include support for configuring various advanced settings for shortcuts that are not configurable in the Express edition. For example, the Premier and Professional editions let you prevent a shortcut in your project from being pinned to the taskbar or to the Start menu. They also let you prevent a shortcut on the Start menu from being highlighted as newly installed after end users install your product. These shortcut options are often used for tools or secondary products that are part of an installation.
- **Merge module authoring and editing**—Package pieces of a project for reuse across application installations. Reuse those you create or any of the ones included in the product. Edit and open modules for greater customization.
- **Project templates**—Create project templates that contain all of the default settings and design elements that you want to use as a starting point when you create an installation project or merge module project.
- **Multiple IIS Web sites**—The Express and Limited editions of InstallShield let you install only one Web site per installation. The Premier and Professional editions let you install more than one Web site per installation.
- **Support for IIS application pools and Web service extensions**—Install and manage IIS application pools and Web service extensions.
- **Advanced support for Windows services**—Although the Express edition of InstallShield includes some support for working with services, the Premier and Professional editions of InstallShield include additional flexibility for services. For example, the Premier and Professional editions enable you to start, stop, or delete a service during installation or uninstallation; the service can be part of your installation, or it can be already present on target systems. These editions also let you configure extended service customization options that were introduced in Windows Installer 5.
- **SQL support**—Connect to SQL servers, import database schema and data, associate SQL scripts with features, and more with SQL support.
- **Ability to modify text files or XML files**—Use the Text File Changes view or the XML File Changes view to configure files that you want to modify on the target system at run time.
- **Pure 64-bit support**—Windows Server Core supports disabling 32-bit Windows-on-Windows (WOW64) support. As this configuration becomes more popular, you may want to ensure that your 64-bit applications can install without any reliance on 32-bit functionality. To make this possible, the Premier and Professional editions of InstallShield include support for building pure 64-bit .msi packages; these can be run on 64-bit Windows-based systems that do not have WOW64 functionality.

- **InstallShield Prerequisite Editor**—Use this tool to create new InstallShield prerequisites and modify existing ones.
- **Custom icons for Setup.exe and Update.exe**—Specify a custom icon (.exe, .dll, or .ico file) that you want to use for `Setup.exe` and `Update.exe` files that you create at build time. The icon is displayed on the Properties dialog box for `Setup.exe`; this Properties dialog box opens when end users right-click the `Setup.exe` file and then click Properties. End users can also see the icon when they view your `Setup.exe` file in Windows Explorer.
- **Expiration dates for Setup.exe**—Set an expiration date, as well as an expiration message, for `Setup.exe`. If end users try to run `Setup.exe` on or after the date that you have specified in your project, the expiration message is displayed, and the installation exits.
- **Support for installation of multiple packages using transaction processing**—Windows Installer 4.5 and later include support for installing multiple packages using transaction processing. The Premier and Professional editions of InstallShield let you add chained .msi packages to an installation project. Your package, plus the added .msi packages, are chained together and processed as a single transaction. If one or more of the packages in the transaction cannot be installed successfully or if the end user cancels the installation, the Windows Installer initiates rollback for all packages to restore the system to its earlier state.
- **Ability to export and reuse various project elements**—Increase efficiency by moving pieces of an existing project (dialogs, custom actions, or features) to a merge module or another installation project.
- **Multiple-instance support**—Create an installation that lets end users install multiple instances of a product on the same machine and in the same user context.
- **Device driver support**—Device driver support in the Premier and Professional editions simplifies the process of installing device drivers from installation using the Driver Installation Frameworks for Applications (DIFxApp) from Microsoft.
- **Additional Dialog Themes**—Several dialog themes are available only in the Premier and Professional editions of InstallShield. The Limited and Express editions contain only two themes.
- **Conversion of Visual Studio merge module projects**—The Premier and Professional editions of InstallShield let you convert a Visual Studio merge module project to an InstallShield merge module project; this is necessary if you want to build a merge module for consumption in other projects.
- **COM+ application proxy support**—Manage COM+ application proxies during your installation. A COM+ application proxy consists of a subset of the attributes of the server application, and it enables remote access from a client machine to the machine where the application resides.

For additional details about the features that are included with each edition, contact InstallShield Sales, or visit http://www.installshield.com.

Converting or Importing Visual Studio Projects into InstallShield Projects

Visual Studio includes limited support for creating setup and merge module projects. InstallShield lets you do the following so that you can use the advanced features and functionality in InstallShield to create installations:

- Import a Visual Studio setup or merge module project (.vdproj) into an Express project (.ise). During the import process, you can choose to import or ignore certain settings in the Visual Studio project.

- Convert a Visual Studio setup project to an Express project.

The import and conversion functionality enables you to create InstallShield installation projects that contain the same data and settings that were in your Visual Studio projects.

Note ▪ *If your Visual Studio project contains one or more project outputs, you can use InstallShield to import that Visual Studio project into an InstallShield project; however, InstallShield cannot convert that Visual Studio project into an InstallShield project.*

Edition ▪ *The Premier and Professional editions additionally let you convert a Visual Studio merge module project to an InstallShield merge module project; this is necessary if you want to build a merge module for consumption in other projects.*

The following versions of Visual Studio are supported:

- Visual Studio 2010
- Visual Studio 2008
- Visual Studio 2005
- Visual Studio .NET 2003
- Visual Studio .NET

Benefits of Using an InstallShield Project Instead of a Visual Studio Project

If you convert your Visual Studio setup project to an InstallShield Express project, you can use the features in InstallShield to customize your project.

Following is a list of a few of the tasks that you can perform in InstallShield projects but not in Visual Studio projects:

- Manage features.
- Manage IIS Web sites, applications, and virtual directories.
- Create DLL, executable file, VBScript, and JScript custom actions that use a source file that is either stored in `Binary` table of the .msi database, or installed with product. Executable file custom actions also support a source file that already exists on the target system. (With Visual Studio, all custom actions must be installed with the product.)
- Add billboards that are displayed during file transfer.
- Create shortcuts to pre-existing files on a target system.
- Manage COM+ applications.

Import Process

InstallShield lets you import a Visual Studio setup or merge module project into an InstallShield Express project.

Note ▪ *If the Visual Studio setup or merge module project that you want to import into an InstallShield project contains one or more project outputs, the InstallShield project must be in the same Visual Studio solution that contains the Visual Studio setup or merge module project and all of its project dependencies.*

In order to import a Visual Studio project that contains project outputs, you must be using InstallShield from within Visual Studio. If your InstallShield project is open in InstallShield, but not from within Visual Studio, and you try to import a Visual Studio project that contains project outputs into the InstallShield project, an error occurs.

Task ***To import a Visual Studio project (.vdproj) into an Express project (.ise):***

1. Create or open the Express project.
2. On the **Project** menu, click the **Visual Studio Deployment Project Wizard** button.
3. Complete the panels of the Visual Studio Deployment Project Wizard.

InstallShield imports the Visual Studio project into your open Express project based on the settings that you configured in the wizard. As InstallShield imports the project, it displays the status of the project import in the Output window. The Output window shows each step of the conversion process, and it lists any conversion errors and warnings. InstallShield imports all of the files, registry entries, and other application data from your Visual Studio project into the Always Install feature of your InstallShield project.

Conversion Process

If you use InstallShield to convert a Visual Studio setup project, InstallShield creates an InstallShield Express project (.ise).

Task ***To convert a Visual Studio project (.vdproj) to an InstallShield project (.ise):***

1. Open InstallShield.
2. On the **File** menu, click **Open**. The **Open** dialog box opens.
3. In the **Files of type** box, select **Visual Studio Setup Projects (*.vdproj)**.
4. Browse to the location of the Visual Studio project that you want to open, and select the project file.
5. Click the **Open** button.

InstallShield creates an InstallShield project based on the settings in the Visual Studio project. InstallShield stores the .ise file in the same folder as the .vdproj file. As InstallShield creates the .ise file, it displays the status of the project conversion in the Output window. The Output window shows each step of the conversion process, and it lists any conversion errors and warnings.

Once the conversion process finishes successfully, the new InstallShield project is displayed in InstallShield.

Post-Import and Post-Conversion Tasks

Prerequisite Tasks

Visual Studio lets you add one or more predefined prerequisites to one or more configurations in a Visual Studio setup project. The import process in InstallShield attempts to convert all of the prerequisites in all of the configurations to equivalent InstallShield prerequisites. If InstallShield does not include a corresponding InstallShield prerequisite, warning -9071 occurs, alerting you that the prerequisite could not be converted.

***Edition** ▪ InstallShield Premier Edition and InstallShield Professional Edition include the InstallShield Prerequisite Editor, a tool that enables you to define custom InstallShield prerequisites and to edit settings for any existing InstallShield prerequisites. If you have either one of these editions of InstallShield, you would be able to create your own InstallShield prerequisites that would be launched during your product's installation.*

In addition, InstallShield Premier Edition and InstallShield Professional Edition enable you to set up release flags to build different versions of an installation—such as Evaluation, Debug, Standard, and Advanced—from a single project. These editions let you include and exclude InstallShield prerequisites at build time, depending on the release flags that you have selected. For example, if you are creating a debug version of your product and you do not want to include some of the InstallShield prerequisites in the build, you can assign a release flag to the appropriate InstallShield prerequisites and then specify which release flags to include in the release.

User-Interface Tasks

The import and conversion processes do not incorporate the dialogs from a Visual Studio project into the InstallShield project. Once you have imported or converted your project, you can use the Dialogs view in InstallShield to configure settings for the dialogs in your project.

Language Tasks

If you imported a Visual Studio project into an InstallShield project and the following conditions exist, InstallShield replaces the existing string entry values in your project with default string entry values for the language of your Visual Studio project:

- You indicate in the Visual Studio Deployment Project Wizard that you would like to import the language of the Visual Studio project.
- The language of your Visual Studio project does not match the language in your InstallShield project. (In Visual Studio, the Localization property indicates the project's language.)

For example, if you indicate in the Visual Studio Deployment Project Wizard that you would like to import the language of the Visual Studio project, if the language of your InstallShield project is Spanish, and if the language of your Visual Studio project is German, InstallShield replaces the Spanish run-time strings in your project with the default German translations. Thus, if you edit a string entry value by revising a setting such as the Publisher setting in the General Information view, and then you indicate in the wizard that you want to import the language of a Visual Studio project, InstallShield overwrites the value of the Publisher setting—as well as values for other settings—with the default German string entry values.

Therefore, if you change the project language while importing your Visual Studio project, review the settings in the General Information view and the Text and Messages view, and modify the string entry values if appropriate.

.NET Installer Class Tasks

If your Visual Studio project includes a .NET installer class custom action, InstallShield configures the .NET installer class information for the .NET assembly during the conversion process. (InstallShield selects the Installer Class check box on the COM & .NET settings tab of the File Properties dialog box to indicate that the assembly's Install, Commit, Rollback, and Uninstall methods will be called at the appropriate times at run time.) InstallShield does not include support for the Condition property of a .NET installer class custom action in Visual Studio. Therefore, if your Visual Studio project contains a .NET installer class custom action that has a condition, you may want to use the Features view in InstallShield to create a condition for the feature that contains the .NET assembly after you have converted your project.

Additional Tasks

You can also use the other views in InstallShield to make additional changes to your project.

Note ▪ *Visual Studio lets you specify a directory path that contains multiple formatted properties, such as [ProgramFilesFolder][Manufacturer]\[ProductName], for the application folder. Visual Studio projects use a directory custom action to resolve the path at run time. However, InstallShield does not support this type of directory path. Therefore, InstallShield resolves the path during the import and conversion processes and uses the* `INSTALLDIR` *property for the path.*

Obtaining Updates for InstallShield

If you have an Internet connection, you can view the updates and obtain the latest InstallShield prerequisites, merge modules, and objects; service packs; patches; and other updates for the version of InstallShield that you are using.

Task ***To check for updates:***

InstallShield shows an update dialog. When an update is available, you can do the following:

- View the updates for your system.
- View description and release notes about the updates.
- Download and Install the updates.

Run-Time Language Support in InstallShield

Edition ▪ *Language support varies, depending on the edition of InstallShield that you are using.*

You can create an installation that runs in a particular language; that is, the run-time dialogs that appear during installation contain text in the specified language.

Creating an installation for one language is the first step in completing your installation project when you plan on distributing your installation in a number of different languages. When you create your project, you can select one of the supported languages from the Project Language list on the New Project dialog box. Your selection determines the language in which all of your installation's run-time dialogs appear.

You can create different installation projects for each language in which you want to distribute your application. If you plan to distribute your installation in both English and German, for example, you can create two different installation projects—one in which you select English as the project language and one in which you select German as the project language.

If you want your installation to appear in a different language, you can export the string entries, translate the strings, and then import them back into your project.

***Note** ▪ You cannot change the language after you have created your project, except by exporting and importing the string entries for translation.*

After you have created your project, you can use the Text and Messages view to edit the text that appears in the run-time dialogs.

***Edition** ▪ The Express project type available in InstallShield supports only one language per installation project. If you want to create an installation that runs in more than one language, it is recommended that you upgrade to the Premier edition of InstallShield; this edition has built-in support for creating projects that run in more than one language. It also offers granular control at both build time and run time over which files are installed based on the language. In addition, the Premier edition enables you to add unsupported languages, beyond the built-in languages, to projects.*

Supported Run-Time Languages

***Edition** ▪ Language support varies, depending on the edition of InstallShield that you are using.*

Note that support for Arabic (Saudi Arabia) and Hebrew is available only in the Premier edition.

The following table lists the run-time languages that InstallShield supports; it also shows the language identifier, or LCID, for each language. The identifier is an integer value that identifies a specific language.

Table 2-4 ▪ Language Identifiers

Language	Identifier
Basque	1069
Bulgarian	1026
Catalan	1027
Chinese (Simplified)	2052

Table 2-4 ▪ Language Identifiers (cont.)

Language	Identifier
Chinese (Traditional)	1028
Croatian	1050
Czech	1029
Danish	1030
Dutch	1043
English (United States)	1033
Finnish	1035
French (Canada)	3084
French (France)	1036
German	1031
Greek	1032
Hungarian	1038
Indonesian	1057
Italian	1040
Japanese	1041
Korean	1042
Norwegian	1044
Polish	1045
Portuguese (Brazil)	1046
Portuguese (Portugal)	2070
Romanian	1048
Russian	1049
Serbian (Cyrillic)	3098
Slovak	1051

Table 2-4 ▪ Language Identifiers (cont.)

Language	Identifier
Slovenian	1060
Spanish	1034
Swedish	1053
Thai	1054
Turkish	1055

Edition ▪ *The Express project type available in InstallShield supports only one language per installation project. If you want to create an installation that runs in more than one language, it is recommended that you upgrade to the Premier edition of InstallShield; this edition has built-in support for creating projects that run in more than one language. It also offers granular control at both build time and run time over which files are installed based on the language. In addition, the Premier edition enables you to add unsupported languages, beyond the built-in languages, to projects.*

Supported Application Programming Languages

InstallShield allows you to create installations for applications created in *any* programming language—including applications created with Java, Pascal, C++, Visual Basic, Delphi, C# .NET, Visual Basic .NET, ASP .NET, and Cobol.

Even if you created your application using a language other than those listed above, you can use InstallShield to create an installation for your application. In addition—if you are not deploying an application, but want to package files or a database—you can use InstallShield to assist with those tasks.

3

Tutorials

This section contains a tutorial that walks you through the process of creating an installation using InstallShield.

Table 3-1 ▪ Available Tutorials

Tutorial	Description
Basic Tutorial	Leads you step-by-step through the process of creating a basic installation, using the InstallShield View List to guide you.

Basic Tutorial

This tutorial is designed to walk you through the process of creating a basic installation, using the InstallShield View List. All of the necessary files are located in the *InstallShield Program Files Folder*\Samples\WindowsInstaller\Basic Installation Project\Data Files subfolder.

The basic tutorial consists of the following tasks:

- Create a New Project
- Organize Your Setup
- Specify Application Data
- Configure the Target System
- Customize the Setup Appearance
- Define Setup Requirements and Actions
- Prepare for Release
- Summary

Create a New Project

Task ***To begin the tutorial:***

1. Launch InstallShield.
2. On the **File** menu, click **New**. The **New Project** dialog box opens.
3. In the **Project Name** box, type **Othello.ise**.
4. In the **Project Language** box, select **English (United States)**.
5. Click **OK**.

The InstallShield Installation Development Environment (IDE) opens.

Organize Your Setup

The first step in building your project is to configure organizational information. This is accomplished through the views associated with the Organize Your Setup step in the View List.

General Information

The General Information view is where you enter information about your application, from your company's name to the Web site your customers can access to learn more about your application.

Task ***To configure general information:***

1. In the View List under **Organize Your Setup**, click **General Information**.
2. In the **Product Name** setting, enter:

   ```
   Othello
   ```

3. In the **INSTALLDIR** setting, enter:

   ```
   [ProgramFilesFolder]Othello
   ```

4. In the **Subject** setting, enter:

   ```
   Othello
   ```

5. In the **Author** setting, enter your name.
6. In the **Summary Information Stream Comments** setting, enter:

   ```
   This installation contains the logic and data required to install My Sample.
   ```

7. Provide an icon for the package:
   a. Click the **Display Icon** setting, and then click the ellipsis button (**...**). The **Change Icon** dialog box opens.
   b. Click the **Browse** button.
   c. Navigate to the following directory:

      InstallShield Program Files Folder\Samples\WindowsInstaller\Basic Installation Project\Data Files

   d. Select `Othello.exe`.
   e. Click the **Open** button. The **Open** dialog box closes, and InstallShield adds the path and name of the file that you selected to the **File name** setting.
   f. Click **OK**.

Features

The next step in organizing your installation is to create features. These features are used to install files and data at run time. InstallShield automatically creates one feature for you: Always Install. This is the only required feature for this project. However, you need to configure it.

Task ***To configure the Always Install feature:***

1. In the View List under **Organize Your Setup**, click **Features**.
2. In the **Description** setting, enter:

   ```
   The executable for the Othello game.
   ```

3. In the **Required** setting, select **Yes**.
4. In the **Visible** setting, select **Not Visible**.
5. In the **Advertised** setting, select **Disallow Advertise**.

6. In the **Comments** setting, enter:

   ```
   This is the only feature.
   ```

Setup Types

In the Setup Types view, you can configure the setup types that the end user can choose from during installation. For this installation, you need only one installation type.

Task ***To configure the setup types:***

1. In the View List under **Organize Your Setup**, click **Setup Types**.
2. Select the **&Typical** check box.
3. Clear the **Minimal** and **Custom** check boxes.

Note ▪ *For this tutorial, you can ignore the remaining views in this step because they are irrelevant to this project. For your own projects, you may want to explore these views to see if you need to use them.*

Specify Application Data

The next step in configuring this basic setup is to specify application data. For purposes of this tutorial, this involves adding files to the already-defined feature.

Files

Task ***To add files to the existing feature:***

1. In the View List under **Specify Application Data**, click **Files**.
2. In the top portion of this view, navigate to the *InstallShield Program Files Folder*\Samples\WindowsInstaller\Basic Installation Project\Data Files folder and click that folder to open it. The files appear in the **Source computer's files** pane.
3. In the **Destination computer's folders** pane, find the folder that contains **[INSTALLDIR]** as part of the folder name.
4. Drag Othello.exe and the three .gif files from the **Source computer's files** pane to the [INSTALLDIR] folder in the **Destination computer's folders** pane.

Configure the Target System

Target system configuration includes creating shortcuts, modifying the registry, configuring ODBC resources, making changes to .ini files, configuring file extensions, and setting environment variables. However, for purposes of this tutorial, you simply need to create a shortcut.

Shortcuts/Folders

The Shortcuts/Folders view lets you create application shortcuts on the target system.

Task ***To create a shortcut for your product:***

1. In the View List under **Configure the Target System**, click **Shortcuts/Folders**.
2. In the **Shortcuts** explorer, right-click **Programs Menu**, and click **New Shortcut**. The **Browse for Shortcut Target** dialog box opens.
3. Browse to the target destination of the `Othello.exe` file that you added to your project in the Files view.
4. Name your shortcut **`Othello`**.
5. In the **Description** setting, enter:

   ```
   Shortcut to the Othello game
   ```

6. In the **Working Directory** setting, select **`[INSTALLDIR]`**.
7. In the **Run** setting, select **Normal Window**.
8. In the **Feature** setting, select **Always Install**.

Note ▪ *For this tutorial, you can ignore the remaining views in this step because they are irrelevant to this project. For your own projects, you may want to explore these views to see if you need to use them.*

Customize the Setup Appearance

Customizing the setup appearance involves configuring which dialogs should appear to the end user during installation, how these dialogs appear, whether billboards are displayed during installation, and what text is displayed. For purposes of this tutorial, you need only to configure the dialogs.

Dialogs

The Dialogs view lets you select and configure the end-user dialogs that compose the user interface for your installation. For this tutorial, you can leave the default values for the Global Dialog Settings properties.

Task ***To customize the dialogs for your end users:***

1. In the View List under **Customize the Setup Appearance**, click **Dialogs**.
2. In the **Dialogs** explorer, clear the **License Agreement** check box and select the **Destination Folder** check box.
3. Select the **Destination Folder** dialog, and then for its **Show Change Destination** setting, select **Yes**.

Note ▪ *For this tutorial, you can ignore the remaining views in this step because they are irrelevant to this project. For your own projects, you may want to explore these views to see if you need to use them.*

Define Setup Requirements and Actions

The views associated with the Define Setup Requirements and Actions View List step enable you to specify criteria required for your application's installation, to add functionality not inherently supported by Windows Installer, and to add files necessary for use during installation. For purposes of this tutorial, you need only to define minimum requirements for installation.

Requirements

In the Requirements view, you can define the target system requirements. If these requirements are not met, your product cannot be installed on the end user's system. The Othello application is fairly simple, but it does require a Pentium processor.

Task ***To define requirements for your installation:***

1. In the View List under **Define Setup Requirements and Actions**, click **Requirements**.
2. In the **Processor** setting, select **Pentium or higher**.

Note ▪ *For this tutorial, you can ignore the remaining views in this step because they are irrelevant to this project. For your own projects, you may want to explore these views to see if you need to use them.*

Prepare for Release

The Prepare for Release View List step enables you to configure, build, test, and distribute your distribution media for your end users.

Build Your Release

The Releases view is where you compile your installation information into an .msi file.

Task ***To build this tutorial project for release:***

1. In the View List under **Prepare for Release**, click **Releases**.
2. In the **Builds** explorer, right-click **CD_ROM** and click **Build**.

As the build begins, the Output window opens at the bottom of the InstallShield interface to display the progress of the build and any error messages.

When the build process is finished, you can find your installation files in a subfolder of the folder where your Othello project is located.

Tip ▪ *Although this tutorial covers building a CD_ROM release, there are several other options for media types. For example, the WebDeployment type enables you to create an installation that is specifically designed for deployment over the Web.*

Test Your Release

The Releases view also enables you to preview your installation program to ensure it functions as expected.

Task ***To test your release:***

1. In the View List under **Prepare for Release**, click **Releases**.
2. In the **Builds** explorer, right-click **CD_ROM** and click **Run Setup**. Your installation launches to install the Othello program.
3. Follow the instructions that the installation program provides. This is the same run-time experience that end users would have if you distributed this installation program to your customers.
4. When the installation is complete, verify that your files and data were installed correctly.
5. Navigate to `C:\Program Files\Othello` (or the folder to which you chose to install the program) and verify that `Othello.exe` is there.
6. Ensure that there is a shortcut called **Othello** on your desktop. Double-click this shortcut to verify that it was configured correctly.

Distribute Your Release

The final step in creating this installation is to distribute it to your target media. Because this is a tutorial and not an installation for an application that you want to distribute to your customers, this step in the tutorial is optional.

Task ***To distribute your release:***

1. In the View List under **Prepare for Release**, click **Releases**.
2. Ensure that you have a CD-ROM drive with a blank CD-ROM in the drive.
3. In the **Builds** explorer, click **CD_ROM**.
4. Click the **Events** tab.
5. Click the **Copy to Folder** setting. InstallShield displays an ellipsis button (**...**) at the right of this setting.
6. Click the ellipsis button (**...**). The **Browse for Folder** dialog box opens.
7. Browse to your CD-ROM drive and click **OK**.
8. In the **Builds** explorer, right-click **CD_ROM** and click **Distribute**.

InstallShield copies all of the necessary installation files to this location—if your CD-ROM drive supports direct copying to CD-ROMs.

Summary

Now that you have finished the tutorial, you can apply what you have learned here to your own installation projects. Most installation projects require the steps and concepts described in this tutorial.

4

Creating Installations

If you have ever installed an application onto your computer, you have seen an installation in action—from the end user's perspective. An installation's primary task is to transfer files from the source medium to the local drive. An installation often also displays a user interface to obtain end user selections, configures the target system (for example, makes any required registry entries and creates shortcuts), and enables modification or uninstallation of the installed application. Creating an installation involves performing some or all of the following tasks.

Specify Installation Information

Basic information that you enter in the General Information view is used in various parts of the installation.

Organize and Transfer Files

File transfer involves copying files from the source medium, such as a CD or DVD, to a local drive on the end user's machine. Depending on the configuration the end user chooses—by selecting a setup type or features—all or only some of the files may be transferred to the local disk.

Organize the files to be installed into setup types and features to help your end users select the most appropriate files.

Configure the Target System

In addition to installing files, many installations need to configure the target system by creating shortcuts and program folders, modifying the registry, modifying initialization file (.ini file) data, configuring Open Database Connectivity (ODBC) resources, modifying environment variables, or installing and starting Windows Services.

Customize Installation Behavior

InstallShield offers wide-ranging customization options. The installations can use custom actions to call .dll functions, launch executable files, run VBScript code, or run JScript code.

Define the End-User Interface

An installation's end-user interface provides information and installation configuration options to the end user. Through the user interface, an end user can choose to install only part of a product, choose to leave some files on the source medium, view a license agreement, or provide to the installer information that may be necessary to ensure the proper configuration of the installation.

The user interface can be customized to meet the needs of your installation. For example, you can prompt a user for a serial number before starting the installation to protect your software against illegal use. During file transfer, an installation can display billboards that provide product information such as new features or usability tips. A status bar may also be displayed to show the progress of the file transfer process.

Configure Servers

A server-side installation may need to create and manage new Internet Information Services (IIS) Web sites or manage COM+ applications and components.

Prepare Installations for Update Notification via FlexNet Connect

FlexNet Connect lets you automatically notify your Web-connected end users when patches, updates, and product information for your application are ready for release. To take advantage of FlexNet Connect, you must enable FlexNet Connect in your original installation.

Prepare the Installation for Maintenance and Uninstallation

To uninstall, modify, or repair an application, the operating system must have some indication that the application is present. To accommodate this, an installation registers an application with the operating system so that it can be easily maintained or uninstalled.

Much of the information registered in this process is available to the end user through Add or Remove Programs in the Control Panel. For example, technical support contact information, product update information, product version, and product publisher information are registered in this process.

Build, Test, and Deploy the Installation

Once you have created your installation project, you will want to build, test, and deploy the installation: create the files that you will release to your users, test the installation for errors, and optionally copy the files to a local or network location or an FTP site.

Before You Begin

The "Before You Begin" section of the InstallShield Help Library contains information that is helpful to installation authors as they create new installation projects with InstallShield. The topics provide background information on Windows Installer, the Windows logo program, and other areas of installation development.

Introduction to Windows Installer

A Windows Installer installation program is distributed as an .msi package, which consists of a Windows Installer database (.msi database) and related data files (.cab files, uncompressed data files, etc.). The .msi databases, implemented as COM structured storage, contain dozens of tables that describe the changes that are to be performed on the target system. For example, some of the .msi tables are:

- `File`, which describes the files to be installed
- `Registry`, which describes the registry data to be written
- `Shortcut`, which describes shortcut settings

Other .msi database tables describe the appearance and behavior of the installation's user interface, install and configure Windows services and ODBC information, determine characteristics of the target system, and store icons and other binary data for use during installation.

From a developer's perspective, perhaps the greatest change in Windows Installer installation programs is that there is no explicit script to write. Instead, Windows Installer–based installations perform standard and custom actions, where an action displays a dialog, queries the target system, or makes changes to the target system. These actions are arranged into sequences, which are ordered collections of actions.

Windows Installer includes a collection of application program interface functions, or APIs, dedicated to managing product installations. Applications must call the Windows Installer APIs in order to take advantage of features available with Windows Installer.

An integral part of Windows operating systems, Windows Installer provides a standard format for component management as well as an interface for managing applications and system tools. Various versions of Windows Installer are available as redistributables for Windows operating systems.

Although it is possible to create a Windows Installer package by editing .msi database tables directly, the large number of tables and relationships among them makes doing so a formidable task. InstallShield organizes the process of developing an installation for Windows Installer into various views, providing graphical editors and wizards that shield the developer from much of the implementation detail that is associated with .msi databases.

Minimizing the Number of User Account Control Prompts During Installation

One of the goals of Windows Vista and later, as well as User Account Control (UAC), is to allow users to run as standard users all of the time. Elevation should rarely be required; if it does occur, it should occur for as short of a duration as possible.

Several different areas of InstallShield affect whether an installation triggers UAC consent or credential prompts for elevated privileges. Understanding these different settings will help you create the appropriate UAC experience for your installation when end users run it on Windows Vista and later systems. It will also enable you to try to minimize the number of UAC prompts that are displayed during your installation.

Depending on how it is configured, an installation that includes InstallShield prerequisites may prompt for elevated privileges on Windows Vista and later systems at several different points during the installation:

1. When the end user launches the `Setup.exe` file
2. When the `Setup.exe` file launches a setup prerequisite that requires elevated privileges

3. When the ISInstallPrerequisites custom action relaunches the `Setup.exe` file in feature prerequisite installation mode because one or more of the features being installed has an associated feature prerequisite

   Note that the ISInstallPrerequisites custom action does not verify whether any feature prerequisites require elevated privileges before the prompt for elevated privileges is displayed. In addition, the ISInstallPrerequisites custom action does not check the conditions on any of the feature prerequisites to determine whether the feature prerequisites need to be installed. The prompt for elevated privileges is always displayed.

4. When the Windows Installer begins the Execute sequence of the .msi package

UAC-Related Settings in InstallShield

Following is a list of the InstallShield settings that help determine whether UAC prompts are displayed during an installation on Windows Vista and later systems:

- **Required Execution Level**—Use this setting on the Setup.exe tab in the Releases view to specify the minimum execution level required by your installation's `Setup.exe` file. InstallShield uses the value that you select (Administrator, Highest available, or Invoker) in the application manifest that it embeds in the `Setup.exe` launcher. For more information, see Specifying the Required Execution Level for Your Setup Launcher on Windows Vista and Later Platforms.

- **Requires Administrative Privileges (for an InstallShield prerequisite)**—The Details pane for InstallShield prerequisites in the Redistributables view has this read-only setting that indicates whether administrative privileges are required in order to install the prerequisite.

- **Require Administrative Privileges (General Information view)**—Use this setting in the General Information view to specify whether the Execute sequence of your installation's .msi package requires administrative privileges. If you set this to No, InstallShield sets bit 3 in the Word Count Summary property to indicate that elevated privileges are not required to install your product. For more information, see General Information View.

In addition, the type of InstallShield prerequisite—either setup prerequisite or feature prerequisite—may affect whether UAC prompts are displayed during an installation on Windows Vista and later systems. To learn more about these two types of InstallShield prerequisites, see Setup Prerequisites vs. Feature Prerequisites.

Note the following UAC-related behavior on Windows Vista and later:

- If Required Execution Level is set to Invoker, any InstallShield prerequisites in your installation do not require administrative privileges, and Require Administrative Privileges in the General Information view is set to No, end users should see no UAC prompts during installation.

- If Required Execution Level is set to Invoker, your installation includes setup prerequisites that require administrative privileges, and Require Administrative Privileges in the General Information view is set to No, end users should see one UAC prompt—plus up to one additional UAC prompt for each reboot—during installation.

- If the full user interface of the setup launcher is displayed and the installation includes setup prerequisites that need to be installed, the setup launcher typically displays the setup prerequisite dialog before the main installation starts. If one or more of the setup prerequisites that need to be installed require administrative privileges, the Install button on the message box has the shield icon to alert the end user that elevated privileges are required.

- If the installation is continuing after a reboot and privileges must be elevated, the OK button of the continuation message box has the shield icon. If privileges do not need to be elevated, the shield button is not displayed.
- If your installation includes more than one setup prerequisite that must be installed on a target machine and one or more of those setup prerequisites requires administrative privileges, the UAC prompt is displayed before the first setup prerequisite is installed. This may allow elevated privileges to be used for all prerequisites without requiring separate UAC prompts for each prerequisite installation. Note, however, that if a setup prerequisite installation causes a reboot, administrative privileges are lost, and a UAC prompt may be displayed if any of the remaining prerequisites require administrative privileges.
- If your installation includes more than one setup prerequisite that must be installed on a target machine and one or more of those setup prerequisites requires administrative privileges, the UAC prompt is displayed before the first setup prerequisite is installed. This may allow elevated privileges to be used for all prerequisites without requiring separate UAC prompts for each prerequisite installation. Note, however, that if a setup prerequisite installation causes a reboot, administrative privileges are lost, and a UAC prompt may be displayed if any of the remaining prerequisites require administrative privileges.

  A slightly different behavior applies to feature prerequisites. If your installation is going to install any features that are associated with prerequisites, the UAC prompt is displayed when the ISInstallPrerequisites custom action relaunches `Setup.exe` in feature prerequisite installation mode. This occurs regardless of whether any of the feature prerequisites require elevated privileges. It also occurs before any of the feature prerequisites' conditions are evaluated to determine whether the feature prerequisites need to be installed. Note that if a feature prerequisite installation causes a reboot, administrative privileges are lost. After the reboot, the ReadyToInstall dialog is displayed again, and the end user needs to click the Install button to proceed with the rest of the installation. In this case, the UAC prompt is displayed again when the ISInstallPrerequisites custom action relaunches `Setup.exe` in feature prerequisite installation mode.
- Note that if Require Administrative Privileges in the General Information view is set to No but your .msi package tries to perform a task for which it does not have adequate privileges, Windows Installer may display a run-time error.
- If privileges are elevated at the end of an installation and the Setup Complete Success dialog launches the product, elevated privileges are carried over to your product. In most cases, running an application with elevated privileges is discouraged.

Example: UAC Prompt Is Displayed for Setup.exe and After a Reboot for a Prerequisite that Requires Administrative Privileges

The following diagram shows when Windows Vista and later request elevated privileges for standard users or administrative users who have limited privileges. The example is based on the default UAC settings on Windows Vista and later systems. The diagram shows an installation that requires elevation for `Setup.exe`, two setup prerequisites, a feature prerequisite, and the Execute sequence of the .msi package. Because the `Setup.exe` file has a manifest that specifies Administrator as the required execution level, elevated privileges are used for each part of the installation. The second UAC prompt is displayed because elevated privileges are lost during a reboot.

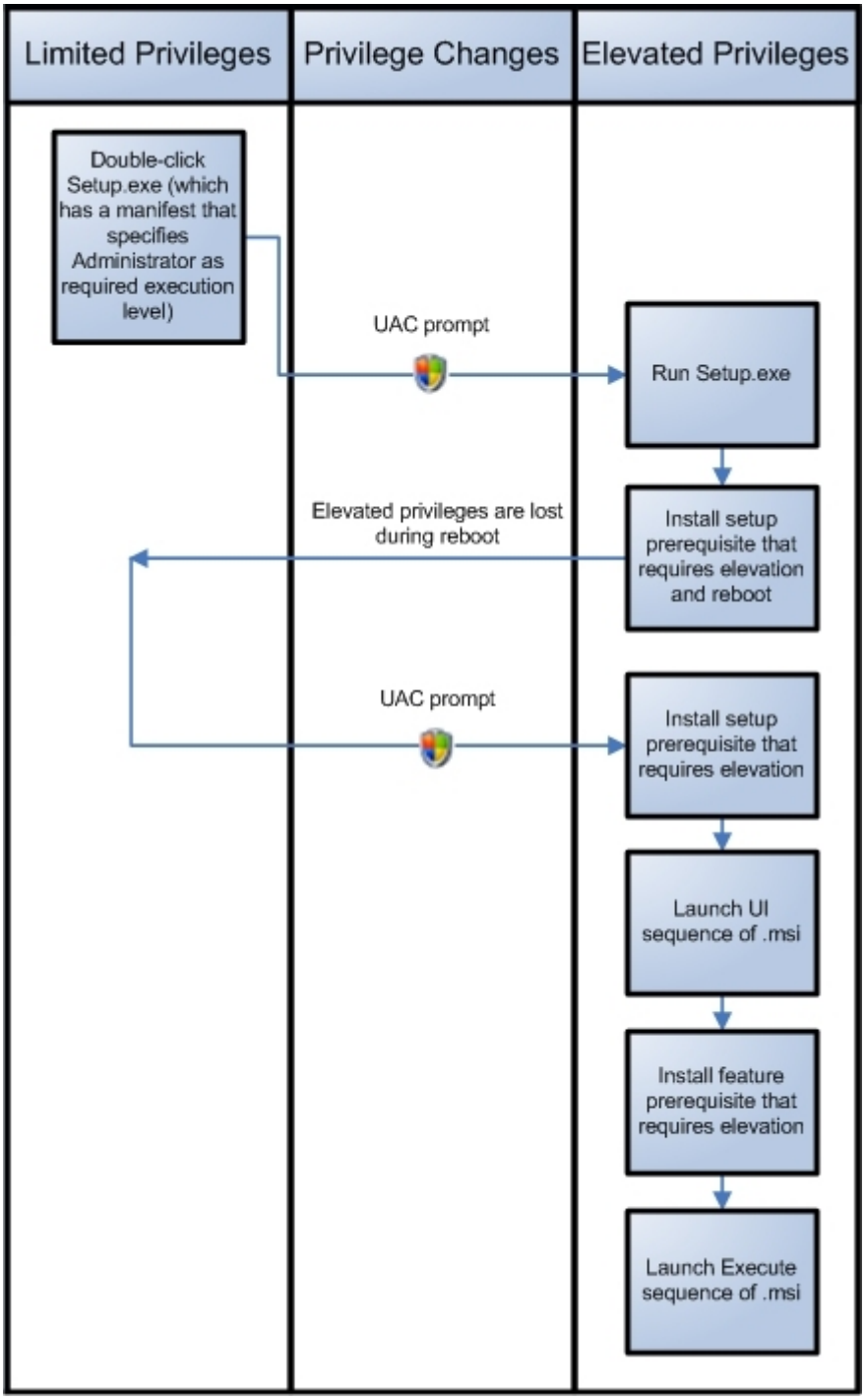


Figure 4-1: Diagram of an Installation that Has Administrator as the Required Execution Level

Requirements for the Windows Logo Program

Microsoft established a list of requirements that a product and its installation must fulfill in order to participate in the Windows 8 Desktop App Certification Program. The requirements outline criteria that help make a product more compatible, reliable, and secure when running on Windows systems. Products that meet the Windows 8 Desktop App Certification Program requirements can carry the Compatible with Windows 8 logo.

To learn how to qualify for the Windows logo program, visit MSDN. This Web site has information about the Windows logo program.

***Edition** ▪ InstallShield Premier Edition and InstallShield Professional Edition include Windows validation suites that may help you identify whether your product meets installation requirements for Microsoft's Windows logo program. If a package or merge module fails one or more validation rules, InstallShield reports the specific rules that were violated and offers additional information to help you troubleshoot the problem.*

***Windows Logo** ▪ The Windows Logo Guideline alert appears throughout the InstallShield Help Library whenever the information relates to complying with the Windows logo program guidelines.*

Preventing the Current Installation from Overwriting a Future Major Version of the Same Product

***Project** ▪ This information applies to Express projects.*

InstallShield includes functionality that lets you prevent the current installation from overwriting a future major version of your product. All new Express projects have this functionality by default.

***Windows Logo** ▪ According to one of the requirements of the Windows logo program, an installation package must prevent end users from installing an earlier version of the product over a later version.*

Task ***To add support for preventing end users from being able to install the current version of your product over a future major version:***

1. In the View List under **Organize Your Setups**, click **Upgrade Paths**.
2. Right-click the **Upgrade Paths** explorer and then click **Prevent Downgrades**.

InstallShield adds an ISPreventDowngrade item to the Upgrade Paths explorer. If end users try to install this version of your product over a newer version of your product, an error message is displayed, informing end users that a newer version is already installed. When end users close the error message box, the installation exits.

***Tip** ▪ To remove the downgrade prevention functionality from your project, right-click the ISPreventDowngrade item in the Upgrade Paths view and then click Delete.*

Preparing Installations for Non-Administrator Patches

Windows Installer 3.0 and later enables you to create patches that can be installed by non-administrators. Non-administrator patches can be used if strict criteria are met. For example, the base installation that the patch will update must include the certificate that will be used to sign the patch package. To learn about the other criteria that must be met, see Non-Administrator Patches.

Task ***To prepare a base installation that can later be updated by non-administrator QuickPatch packages:***

1. In the View List under **Prepare for Releases**, click **Releases**.
2. In the **Builds** explorer, select the media type that you want to create or modify.
3. Click the **Signing** tab.
4. Specify the digital signature information.

InstallShield adds the necessary information to the `MsiDigitalCertificate` table and the `MsiPatchCertificate` table. This enables you to create QuickPatch packages that can be installed by non-administrators.

Specifying Installation Information

As you begin creating your installation project, you will need to specify important installation information at the outset. Installation information can be defined by the high-level properties and considerations you need to set and determine about the organization of your installation. This includes specifying product and project properties and configuring Add or Remove Programs properties.

Configuring General Project Settings

InstallShield stores your project settings in a single installation project file (.ise file). This file stores all of the information about your project. In the General Information view, you can edit basic information about your installation project—including the author's name, the languages that the project supports, and any comments that you want to include.

The General Information view is also where you configure general product information such as the product name, the product code (GUID), and the version number. A product is the top level of organization in an installation project. The installation is further divided into features, which are subsets of your product.

For detailed information about each of the project and projects settings that are available, see General Information View.

Setting the Product Code

The product code is a string that uniquely identifies your product. An installation uses a product code at run time to determine whether the product has already been installed.

Enter a GUID that uniquely identifies your product, or click the **Generate a new GUID** button ({...}) in the Product Code setting in the General Information view to let InstallShield generate a new GUID. The installation registers this GUID at run time.

Caution ▪ *Because the product code uniquely identifies your product, changing the code after distributing a release of your product is not recommended.*

Specifying a Product Name

Enter the name of your product in the Product Name setting of the General Information view. The name that you enter should be the name of the product for which you are creating an installation. This value is used throughout your project, in various instances; for example:

- The name of the source file folder under the project location
- The name of the Windows Installer package (.msi file) that InstallShield builds
- In run-time dialogs
- Under the informational registry key in accordance with Windows logo requirements. The informational values are found in the following location and are used in Add or Remove Programs to enable an end user to change or remove your product.

HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Uninstall\<product code>

Since it will be incorporated into the paths for your source files, the product name cannot contain any of the following characters: \ / : * ? " < > | . -

Tip ▪ *If you want to include an ampersand (&) in your product name, you must use two ampersands (&&) to display the name properly in end-user dialogs. For example, to display* **New & Improved Product**, *you should enter the product name as* `New && Improved Product`.

Specifying the Product Version

When you are specifying the version number for your product, you must ensure that you enter a valid product version. The version must contain only numbers. It is typically in the format *aaa.bbb.ccccc* or *aaa.bbb.ccccc.ddddd*, where *aaa* represents the major version number, *bbb* represents the minor version number, *ccccc* represents the build number, and *ddddd* represents the revision number. The maximum value for the *aaa* and *bbb* portions is 255. The maximum value for *ccccc* and *ddddd* is 65,535.

At run time, the installation registers the version number of the product that is being installed. The entire version string is displayed in Add or Remove Programs. The product version number is important because the installation engine uses it in part to determine whether to apply an upgrade.

You can configure the product version in the General Information view.

Note that although you can include the fourth field (*ddddd*) when you specify your product's version, the installation does not use this part of the product version to distinguish between different product versions.

If your release includes a `Setup.exe` file, the product version that you specify is displayed on the Properties dialog box for `Setup.exe`. For more information, see Customizing File Properties for the Setup Launcher.

Setting the Upgrade Code

The upgrade code is a GUID that identifies a related set of products. The Windows Installer uses a product's upgrade code when performing major upgrades of an installed product. The upgrade code, stored in the `UpgradeCode` property, should remain the same for all versions of a product.

To configure the GUID for a new product, use the Upgrade Code setting in the General Information view. When you are working on an upgrade for a later version of your product, enter the same upgrade GUID in the Upgrade Paths view or in your QuickPatch project.

Configuring Add or Remove Programs Information

The settings in the Add or Remove Programs area of the General Information view describe the information that appears in Add or Remove Programs tool in the Control Panel.

If you have not entered a value for a particular Add or Remove Programs setting, a sample value appears in grey text. This value is meant to serve as an example and is not included in the final Windows Installer database.

Entering Summary Information Stream Data

The Summary Information Stream contains information about your company and the software being installed. This information can be accessed by right-clicking an .msi file and selecting Properties. Then, click the Summary tab to view the summary information.

Setting the Summary Information Stream Comments at Build Time

You also have the option of entering comments at build time. You can use the `SummaryInfoComments` parameter of `MSBuild.exe` task to set the **Summary Information Stream** comments at build time, such as including the build number, as shown in the following example:

```
MSBuild.exe c:\installers\Setup.sln /Property:SummaryInfoComments="Insert Comments Here"
```

Setting the Default Product Destination Folder (INSTALLDIR)

Your project's `INSTALLDIR` property serves as the default folder for all of your product's files. Its value is assigned to the Windows Installer folder property `INSTALLDIR`, which is the default feature destination folder.

Windows Logo ▪ *According to the Windows logo program requirements, the default destination of your product's files must be a subfolder of the Program Files folder or the end user's Application Data folder, regardless of the language of the target system. If you use ProgramFilesFolder as the parent folder for your product's destination folder setting, your files are installed to the correct location.*

The default value for `INSTALLDIR` is:

`[ProgramFilesFolder]`*`Company Name\Product Name`*

Task ***To set a product's INSTALLDIR property:***

1. In the View List under **Organize Your Setup**, click **General Information**.
2. To use a built-in Windows Installer directory as part of your path: In the **INSTALLDIR** setting, click the ellipsis button (...) . The **Set INSTALLDIR** dialog box opens. In the **Destination Directories** box, select a destination folder.

   As an alternative, you can manually enter the path in the **INSTALLDIR** setting.

Note ▪ *Selecting a new folder property in the Set INSTALLDIR dialog box overwrites the contents of the value in the INSTALLDIR setting. You can specify a subfolder of any folder property by separating subfolders with a backslash—for example,* **[ProgramFilesFolder]My Company\Program***.*

When specifying a destination folder, you cannot include a space after the closing square bracket (]), or before or after a backslash (\). For example, the following paths are not valid:

`[ProgramFilesFolder] \YourCompany\default`

`[ProgramFilesFolder]YourCompany\ default`

When you use an installer folder property such as INSTALLDIR, you are specifying a default value. An end user could change this value by setting a property when launching Msiexec.exe at the command line or by selecting a new destination folder for a feature in the Custom Setup dialog.

INSTALLDIR and the Registry

You can set the default installation directory for your installation to be a key value found in the target system's registry. This directory is stored in the INSTALLDIR property in the General Information view.

One limitation of using a registry entry to set the value of INSTALLDIR is the fact that the registry entry must exist on the target system before your installation starts. If the registry entry cannot be found, INSTALLDIR is set to the last valid value you had entered for it.

For example, when you first create an installation project, the value of INSTALLDIR is [ProgramFilesFolder]Your Company Name\Default. If you were to change this to read from the registry, and the registry entry is not found, INSTALLDIR reverts to the previous value of [ProgramFilesFolder]Your Company Name\Default.

Setting INSTALLDIR from the Registry

Task ***To set the INSTALLDIR value from the registry:***

1. In the View List under **Organize Your Setup**, click **General Information**.
2. In the **INSTALLDIR** setting, enter the full path to the registry key containing the value you want to use. For example:

   HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\App Paths\App.exe

Setting this value to **HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\App Paths\App.exe** will instruct the installation to get this value from a string named **App.exe** in the App Paths key. Setting this value to **HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\App Paths\App.exe** (with trailing backslash) will instruct the installation to get this value from the default value of the key named **App.exe**.

Note ▪ *The trailing backslash specifies that the final portion of the specifier is the name of a key and not a value name.*

Securing Files, Folders, and Registry Keys in a Locked-Down Environment

InstallShield offers different ways to secure files, folders, and registry keys for end users who run your product in a locked-down environment:

- **Traditional Windows Installer handling**—InstallShield stores permission information for your product in the LockPermissions table of the .msi database.
- **Custom InstallShield handling**—InstallShield stores permission information for your product in the ISLockPermissions table of the .msi database. InstallShield also adds custom actions to your project.

These methods enable you to assign permissions for a file, folder, or registry key to specific groups and users. For example, you may assign Read, Write, and Delete permissions for a particular file to the Administrators group, but only Read permissions for all of the users in a different group.

Determining Which Option to Use

The following table compares the different types of methods for setting permissions.

Table 4-1 ▪ Comparison of Different Ways to Secure Objects (Files, Folders, and Registry Keys) in a Locked-Down Environment

Comparison Category	Explanation of Available Support
Well-known security identifiers (SIDs)	• **Traditional Windows Installer handling**—Supports a limited number of SIDs (Administrators, Everyone). • **Custom InstallShield handling**—Supports many SIDs (Administrators, Authenticated Users, Creator Owner, Everyone, Guests, Interactive, Local Service, Local System, Network Service, Power Users, Remote Desktop Users, and Users).
Localized names for SIDs	• **Traditional Windows Installer handling**—Does not support localized names for SIDs; if you try to use a localized name, the installation fails. • **Custom InstallShield handling**—Supports localized names for all of the supported well-known SIDs (Administrators, Authenticated Users, Creator Owner, Everyone, Guests, Interactive, Local Service, Local System, Network Service, Power Users, Remote Desktop Users, and Users).
Ability to deny specific permissions	• **Traditional Windows Installer handling**—Not supported. This handling lets you set specific permissions; you cannot deny permissions. Thus, you can give a user read-only access to a file. However, you cannot prevent a user from having read-only access. • **Custom InstallShield handling**—Supported. This option lets you indicate whether you want to deny a user or group from having the permissions that you are specifying.
Effect on permissions that already exist	• **Traditional Windows Installer handling**—Existing permissions may be deleted. For example, if permissions are already set for a folder on the target system for the Everyone user, and your installation needs to set permissions for the Administrators user, this option would allow you to set permissions for the Administrators user. However, the existing permissions for Everyone would be deleted. • **Custom InstallShield handling**—This option lets you add permissions to a file, folder, or registry key that already exists on the target system, without deleting any existing permissions for that object. For example, if permissions are already set for a folder on the target system for the Everyone user, and your installation needs to set permissions for the Administrators user, these options would allow you to set permissions for the Administrators user without deleting the existing permissions for the Everyone user.

Table 4-1 ▪ Comparison of Different Ways to Secure Objects (Files, Folders, and Registry Keys) in a Locked-Down Environment (cont.)

Comparison Category	Explanation of Available Support
Ability to propagate permissions to child objects (subfolders, files, and subkeys)	• **Traditional Windows Installer handling**—Not supported. If you want to configure permissions for a subfolder or a file in a folder (or a subkey under a registry key), the parent that is created on the target system automatically inherits the permissions of its child. • **Custom InstallShield handling**—Supported. This option lets you configure permissions for a folder (or a registry key), and indicate whether you want the permissions to be applied to all of the folder's subfolders and files (or the registry key's subkeys).
Ability to set permissions for a new user that is being created during the installation	• **Traditional Windows Installer handling**—Not supported. • **Custom InstallShield handling**—Supported. If a new user is created during the installation, you can configure permissions for that user.

Neither option lets you set permissions for objects that are not being installed as part of your installation.

Learning More about the Custom InstallShield Handling Option or the Traditional Windows Installer Handling Option

In Express projects, you need to specify whether you want to use the custom InstallShield handling or the Windows Installer handling. To learn how, see Selecting the Locked-Down Permissions Type for a Project.

To learn how to set permissions for a file or folder using either of these options, see Configuring Permissions for Files and Folders. For information on setting permissions for a registry key using either of these options, see Configuring Permissions for Registry Keys.

Selecting the Locked-Down Permissions Type for a Project

InstallShield includes a project-wide setting that lets you specify how your installation should configure permissions for files, folders, and registry keys for end users in a locked-down environment.

Task ***Selecting the locked-down permission type for a project:***

1. In the View List under **Organize Your Setup**, click **General Information**.
2. In the **Locked-Down Permissions** setting, select the appropriate option:
   - **Custom InstallShield handling**—InstallShield adds a custom table and custom actions to your project to set permissions on the target system. This is the default value.
   - **Traditional Windows Installer handling**—InstallShield uses the `LockPermissions` table in the .msi database to store permissions information for your product.

For a detailed comparison of these two options, see Securing Files, Folders, and Registry Keys in a Locked-Down Environment.

The next time that you configure permissions for a file, folder, or registry key in your project, InstallShield uses the locked-down permission type that you selected:

- If you selected the traditional Windows Installer handling option, InstallShield uses the `LockPermissions` table in your project.
- If you selected the custom InstallShield handling option, InstallShield uses the `ISLockPermissions` table in your project; InstallShield also adds the ISLockPermissionsCost and ISLockPermissionsInstall custom actions to your project.

If you change the value of the Locked-Down Permissions setting but your project already contains permission settings for files, folders, or registry keys, InstallShield displays a message box that lets you specify whether you want to migrate the permission data to the appropriate table. If you choose to migrate the data, InstallShield moves the data to the table that corresponds with the option that you selected; if you are switching from the custom InstallShield handling option to the traditional Windows Installer handling option, InstallShield also deletes the ISLockPermissionsCost and ISLockPermissionsInstall custom actions from your project.

Specifying Whether Windows Installer Installations Should Be Logged

InstallShield enables you to specify on a project-wide basis whether Windows Installer 4.0 or later should log your installation. You can also customize the types of messages that are logged.

Task ***To specify project-wide logging information for Windows Installer 4.0 or later:***

1. In the View List under **Organize Your Setup**, click **General Information**.
2. Click the **Create MSI Logs** setting, and then click the ellipsis button (**...**). The **Logging Options for Windows Installer 4.0 and Later** dialog box opens.
3. Select the appropriate option. If you select the custom option, enter the MsiLogging value.
4. Click **OK**.

The results vary, depending on which option you select in the Logging Options for Windows Installer 4.0 and Later dialog box:

- If you select No, installations are not logged. This is the default value.
- If you select Yes, InstallShield populates the `MsiLogging` property with the default value of *`voicewarmupx`*. If the installation is run on a target system that has Windows Installer 4.0, the following occurs:
  - The installer creates a log file according to the default logging mode of *`voicewarmupx`*.
  - The installer populates the `MsiLogFileLocation` property with the log file's path.
  - A **Show the Windows Installer log** check box is added to the SetupCompleteSuccess, SetupCompleteError, and SetupInterrupted dialogs. If the end user selects that check box and then clicks Finish, the log file is opened in a text file viewer or editor.
- If you select Custom, InstallShield populates the `MsiLogging` property with the value that you specify in the box. If the installation is run on a target system that has Windows Installer 4.0, the following occurs:

- The installer creates a log file according to the custom value that you specified in the box.
- The installer populates the `MsiLogFileLocation` property with the log file's path.
- A **Show the Windows Installer log** check box is added to the SetupCompleteSuccess, SetupCompleteError, and SetupInterrupted dialogs. If the end user selects that check box and then clicks Finish, the log file is opened in a text file viewer or editor.

Earlier versions of Windows Installer ignore the MsiLogging setting. The **Show the Windows Installer log** check box is not visible in run-time dialogs that are displayed on systems running earlier versions of Windows Installer.

Important ▪ *The* `MsiLogFileLocation` *property is read-only; it cannot be used to set or change the log file location.*

Including a Software Identification Tag for Your Product

Project ▪ *This information applies to Express projects.*

ISO/IEC 19770-2 is an international standard for the creation of software identification tags. A software identification tag is a small XML-based file that contains descriptive information about the software, such as the product name, product edition, product version, and publisher. Software asset management tools collect the data in the tags to provide accurate application identification for software that is installed in an enterprise.

Software identification tagging is evolving as an industry standard, enabling independent software vendors to create smarter applications that give their customers better information for software asset management and license optimization initiatives. Including the identification tag in your product's installation makes it possible for your customers to use tools that can monitor their internal usage of your product, allowing them to understand, manage, and optimize the number of licenses of your product that they obtain from you.

Proper tag creation requires that you configure standard General Information settings such as Product Name and Product Version. It also requires that you configure a few identification-specific settings, which are also in the General Information view.

Task ***To include a software identification tag in your installation:***

1. In the View List under **Organize Your Setup**, click **General Information**.
2. In the **Software Identification Tag** area of the view, modify the values of the settings as needed.

   The **Use Software Identification Tag** setting lets you specify whether you want to include a tag in your installation. Select **Yes**, which is the default value, and then configure the other settings in the **Software Identification Tag** area as needed.

At build time, if the following conditions are true, InstallShield includes the software identification tag with the installation that it builds:

- Yes, the default value, is selected for the Use Software Identification Tag setting in the General Information view.
- The Unique ID, Tag Creator, and Tag Creator ID settings in the General Information view have values.

Note that if tagging is enabled but you have not entered values in one or more of the three aforementioned tag identification settings, InstallShield generates a build warning to inform you that the tag could not be included in your release. To resolve this warning, configure the settings in the Software Identification Tag area of the General Information view as needed.

When you use tagging in your project, InstallShield adds the tag to the Always Install feature in your project. At run time, the file is installed to two different locations:

- `INSTALLDIR`
- `CommonAppDataFolder`

If you configure your project to include a software identification tag and you also configure the release in the Releases view to use a .pfx file to digitally sign your release, InstallShield digitally signs the tag at build time. Note that the .NET Framework 3.5 must be installed on your build machine in order to sign a tag file.

Organizing Files for Your Installation

The primary task of an installation is to transfer files from your distribution media to your end user's hard drive. In an InstallShield installation, files are organized in a hierarchy: files are included in features (and optionally subfeatures), and features are associated with setup types.

At run time, your end user simply selects the setup type—or, if you allow it, the features and subfeatures—that he or she wishes to install.

A file often relies on functions in other files to perform a task. However, you may not be aware of all these other files—known as dependencies—when you include your application's files in your project. To help you identify dependencies, InstallShield offers three dependency scanners that automatically add these files to your project.

In addition to including individual files in your project, you can also include redistributables (InstallShield prerequisites, merge modules, and objects), which contain logic and files needed to install distinct pieces of functionality. For example, to include the Java Runtime Environment (JRE) files in your installation, you can add the InstallShield prerequisite for JRE to your installation project.

Designing Installations

A feature is the smallest separately installable piece of your product from the end user's standpoint. Individual features are visible to end users when they select a Custom setup type during installation.

Features

Features are the building blocks of an application from the end user's perspective. Each feature represents a specific piece of functionality for your product—such as the help files. End users should be able to install and uninstall discrete features of your product.

For example, an end user with limited hard drive space could elect not to install a product tutorial. If the user subsequently purchases another computer or frees resources on an existing one, the previously uninstalled product tutorial could then be installed.

You should separate your application into features that correspond to functionality of your application and that can be available in or removed from different setup types.

Setup Types

Setup types enable you to allow your end users to selectively install portions of your product or all parts of your product. The default setup types are Typical, Minimal, and Custom.

Setup types are based on features. You select which features you want to associate with each setup type. Then, when an end user selects a certain setup type, only those features you associated with that setup type are installed.

Separating Applications into Features

Separate your application into independent parts, such as help, clip art, and program files. This gives the end user combination options when installing your application. For instance, if your application contains a large, graphic-intensive clip art file, you could make the clip art file a feature. This gives the end user the option of installing or not installing the file—a vital ability for users with limited available resources.

In separating your application into features, make sure that end users can recombine the parts in several ways to fulfill specific needs. When doing this, consider the needs of all users, from system administrator to customer service representative to developer and everyone in between. By addressing all groups of users, you are promoting increased distribution and usage of your application.

Each feature should have one function—such as help files—and should be clearly defined according to this functionality to ensure recognition and comprehension. Features should possess an independent functionality so that users can install and use the feature by itself, if they so choose.

If one feature requires another, make the dependent feature a child of the other.

To eliminate confusion, keep any information pertaining to the management of the system or application transparent to the user.

Defining Features

A feature is a building block of an application from the end user's perspective. It represents a specific capability of your product—such as its help files or a part of a product suite that can be installed or uninstalled based on the end user's selections. Your entire application should be divided into features that perform a specific purpose. You can define features in the Features view.

A feature should be self-contained, in the sense that a feature should not require sibling features. For example, a thesaurus feature should not require a dictionary feature that the end user can choose not to install. However, you can design your installation to contain subfeatures of a "parent" feature, which allow the end user finer control over which files and data to install.

Subfeatures are further divisions of a feature. If all features and subfeatures are visible, your end user can then select which portions of a feature to install in the Custom Setup dialog.

Tip ▪ *Although you can create many levels of subfeatures, you should keep the design as simple as possible for organizational purposes.*

Creating Features

You can use the Features view to create features, as well as subfeatures, for your project.

Task ***To create a feature:***

1. In the View List under **Organize Your Setup**, click **Features**.
2. Right-click the **Features** explorer and click **New Feature**. InstallShield creates a new feature with the default name **NewFeature***n* (where *n* is a successive number).
3. Type a name for the feature. To rename the feature, right-click it and click **Rename**.
4. Configure the feature's settings in the right pane.

Creating Subfeatures

Task ***To create a subfeature:***

1. In the View List under **Organize Your Setup**, click **Features**.
2. In the **Features** explorer, right-click the parent feature that should contain the subfeature and then click **New Feature**. InstallShield creates a new subfeature with the default name **NewFeature***n* (where *n* is a successive number).
3. Type a name for the subfeature. To rename the subfeature, right-click it and click **Rename**.
4. Configure the feature's settings in the right pane.

Tip ▪ *You can create multiple nested features at one time by adding a new feature and typing* `Feature 1\Feature 2\Feature 3` *for the feature's name. InstallShield creates a nested feature structure where* `Feature 3` *is a subfeature of* `Feature 2`*, which is a subfeature of* `Feature 1`.

Configuring Feature Settings

Task ***To configure a feature's settings:***

1. In the View List under **Organize Your Setup**, click **Features**.
2. Select the feature that you want to configure.
3. Configure the settings in the grid in the right pane.

Setting Feature Conditions

Conditional installation of one of your features can be useful if you need to specify that a feature in your installation requires a specific operating system or application. If the target system does not meet the condition, the feature is not installed.

Note ▪ *You cannot create conditions for the Always Install feature.*

Task ***To set a condition for a feature in your project:***

1. In the View List under **Organize Your Setup**, click **Features**.
2. Select the feature that you want to configure.
3. Click the **Condition** setting and then click the ellipsis button (**...**). The **Condition Builder** dialog box opens.
4. To create an operating system condition:

a. Click the **Operating System** tab.

b. Clear the **All Operating Systems** check box and select the operating systems that you want to target in the condition.

To create a software condition:

a. Click the **Software** tab.

b. Do one or more of the following:

- If an application must be present on the target system for the selected feature, set the check box of that application to the green checked state . To set the check box to this state, click it until the green check mark appears in the check box.
- If an application must not be installed on the target system for the selected feature, set the check box of that application to the red X . To set the check box to this state, click it until the red X appears in the check box.
- To remove a software condition, click the check box until it is empty. An empty check box indicates that the condition is not based on the software in any way.
- If the application is not listed, click **Create New Requirement**. The **System Search Wizard** opens, enabling you to create a new condition based on whether a specified file exists on the target system. The new condition is created and added to the **System Software Requirements** explorer.

5. Click **OK**.

InstallShield adds the condition to the grid. The feature is installed if any of the operating system conditions for this feature are true and if all of the other conditions for this feature are also true.

Displaying Features to End Users

InstallShield enables you to indicate how you want a feature to be presented to the end user in the Custom Setup dialog.

The Visible setting for a feature in the Features view is where you indicate whether and how the feature should be displayed. Available options are:

Table 4-1 ▪ Options that Are Available for the Visible Setting

Option	Description
Visible and Collapsed	The feature is displayed in the Custom Setup dialog with its subfeatures collapsed by default.
Visible and Expanded	The feature is displayed in the Custom Setup dialog with its subfeatures expanded by default.
Not Visible	The feature is be displayed to the end user in the Custom Setup dialog.

Note ▪ *Selecting Not Visible for this setting does not have any direct impact on whether a feature is installed. A feature is not automatically installed if it is invisible—it just cannot be deselected if it would otherwise be installed, or selected if it should not be installed.*

Requiring Features to Be Installed

When you select Yes for a feature's Required setting, the end user cannot deselect it in the Custom Setup dialog. The feature will be installed to the target system.

If the value for the Required setting is No, the feature is installed by default, but the end user can deselect it.

Advertising Features

InstallShield enables you to selectively enable or disable a feature for advertisement. Advertised features are not installed immediately during the installation process. Instead, they are installed when requested. If the feature is assigned, the feature appears to be already installed, although it is not installed until the end user requests it. (Assigned features have their shortcuts installed, and they can be installed from Add or Remove Programs in the Control Panel. However, an assigned feature is advertised until a user requests it.) A published feature does not appear on the target system until it is requested from the installer. (Published features lack any end user–interface elements. They are installed programmatically or through an associated MIME type.)

Use the Advertised setting in the Features view to specify whether advertisement should be allowed. Available options for this setting are:

Table 4-2 ▪ Options that Are Available for the Advertisement Setting

Option	Description
Allow Advertise	End users have the ability to select the advertisement option for this feature in the CustomSetup dialog. Although advertisement is allowed, it is not the default option when the installation is run.
Favor Advertise	The feature is advertised by default. End users can change the advertisement option for a feature in the CustomSetup dialog.
Disallow Advertise	Advertising is not allowed for this feature. End users cannot elect to have the feature advertised in the CustomSetup dialog.
Disable Advertise if Not Supported	Advertisement works only on systems with Internet Explorer 4.01 or later. If the target system does not meet this criterion, advertising is not allowed. If the target system can support advertisement, advertising is allowed.

When you allow feature advertisement, the feature is advertised, regardless of the mode in which the installation is running, as long as no other factors prevent it from being advertised. In the Custom Setup dialog, the end user can control which features are immediately installed and which are available later.

Advertisement usually requires support from the application. For example, your product's spell checker can be advertised. The application interface offers use of the spell checker through a menu command or toolbar button. You must write to check the feature's installation state and install it when the customer clicks the Spell Check command or button.

Setting a Feature's Remote Installation Setting

The Remote Installation setting for a feature determines whether the feature's files are installed on the target system or run from the source medium, such as a CD-ROM or network server. The default value for a new feature is Favor Local, which means that the files in the selected feature are installed on the target system.

Task ***To change the Remote Installation setting so that the feature's files run only from the source medium:***

1. In the View List under **Organization**, click **Features**.
2. Select the feature that you want to configure.
3. In the **Remote Installation** setting, select **Favor Source**.

Tip ▪ *Selecting **Favor Parent** gives a subfeature the same value as its parent feature.*

Changing the Feature Order in the Custom Setup Dialog

The order in which the features are listed in the Features view is the order that the features will be displayed on the Custom Setup dialog.

Task ***To change the feature order on the Custom Setup dialog:***

1. In the View List under **Organize Your Setup**, click **Features**.
2. Right-click the feature that you want to move and click **Move Up** or **Move Down**. You can also move a feature left or right, thereby making it a subfeature of another feature.

Tip ▪ *You can also reorder your features through simple dragging and dropping. Any feature or subfeature can be moved in this way.*

Working with Setup Types

Setup types enable you to allow your end users to selectively install portions of your product or all parts of your product. The default setup types are Typical, Minimal, and Custom.

The Typical setup type usually includes most, if not all, of an application's features. The Custom setup type lets the end user determine which features should be installed on the target system. The Minimal setup type usually includes the features that are absolutely necessary to run your application; this type is designed for end users who have limited disk space, such as those using notebook computers.

Setup types are based on features. You select which features you want to associate with each setup type. Then, when an end user selects a certain setup type, only those features you associated with that setup type are installed.

Specifying Features Included in Setup Types

Task ***To specify the features included in each setup type:***

1. Open the **Setup Types** view.
2. In the **Setup Types** explorer, click the setup type that you want to edit. All of the features in your installation are listed in the pane below.
3. Clear the check boxes next to the features that you do not want included in the selected setup type.

Renaming Setup Types

Changing the name in the Setup Types view is equivalent to modifying the resources for the Setup Types dialog in the Text and Messages view.

Task ***To rename a setup type:***

1. Open the **Setup Types** view.
2. In the **Setup Types** explorer, right-click the setup type that you want to rename, and then click **Rename**,

   As an alternative, you can click the setup type and then press F2.
3. Type the new name for the setup type. This name is displayed during installation in the Setup Types dialog.

Note ▪ *To provide an accelerator key for your setup type, include an ampersand (&) before a letter in the name. For example, the name* `Cu&stom` *becomes the label Custom with an underlined s, and the end user can select its option button during setup by pressing the S key. To set the ampersand symbol as an accelerator key, type* `&&`*.*

Specifying Setup Types

Task ***To specify the setup types to include in your installation:***

1. Open the **Setup Types** view.
2. In the **Setup Types** explorer, clear the check boxes for the setup types that you do not want included in your installation.
3. Optionally, type a description in the **Description** property field. To enter a multi-line description, use the \n escape sequence. For example, if you type `InstallShield\nExpress Edition`, the description in the **Setup Types** dialog appears as:

```
InstallShield
Express Edition
```

Including Only One Setup Type in an Installation

If you want all of your end users to have the same files installed, you should include only the Typical setup type in your project. Then when the installation runs, the Setup Type dialog is not displayed, and the end user cannot choose which features should be installed. All features are installed.

Task ***To include only one setup type in an installation:***

1. Open the **Setup Types** view.
2. In the **Setup Types** explorer, clear the **Minimal** and **Custom** check boxes.

The Setup Type dialog is not included in the run time of the installation; therefore, end users cannot selectively install only parts of your product.

Accessing the Setup Type at Run Time

The _IsSetupTypeMin property stores the setup type that the end user selects in the Setup Type dialog.

Task ***To access the selected setup type during the run time of an installation, do one of the following:***

- Use the following VBScript code:

```
' Get the value of the setup type selected
Dim sSetupType
sSetupType= Session.Property("_IsSetupTypeMin")

' Show it.
MsgBox sSetupType
```

The _IsSetupTypeMin property contains only the default setup type names—Typical, Minimal, and Custom. If you rename Minimal in the Setup Types view to MySetupType and the end user selects the renamed setup type, _IsSetupTypeMin will contain Minimal and not MySetupType.

Including Files and Folders

Your files, the core of your product, are also the core of your installation. When you add a file to your project, you must select which feature with which it is associated. Features are what end users see when they run your installation and select the Custom setup type. If the feature is selected for installation, the feature's files are installed to the target system.

You can add files to your project in the Files view.

Adding Files and Folders to a Project

Use the files explorer in the Files view to add folders and files to your project.

Task ***To add files and folders using the files explorer:***

1. In the View list under **Specify Application Data**, click **Files**.
2. In the **Feature** list, select the feature that you want to contain the file or folder that you are adding.
3. In the **Destination computer's folders** pane, select the folder to which you want to add a folder or file.

   `INSTALLDIR` is the most commonly used target location, because this is the default root directory for your application's files.

   If the predefined folder to which you adding the folder or file is not displayed in this pane, you can add it. To learn how, see Displaying Predefined Folders in the Files View.

Tip ▪ *If you are creating a 64-bit installation that can install to 64-bit file locations (such as the Program Files Folder instead of the Program Files Folder (x86) folder) on 64-bit target systems, select a predefined 64-bit folder (ProgramFiles64Folder, CommonFiles64Folder, or System64Folder) or one of its subfolders in the Files view.*

Note that a 64-bit installation cannot be run on 32-bit target systems. To learn more, see Challenges of Supporting Both 32-Bit and 64-Bit Target Operating Systems.

4. To optionally add a custom folder that you want the installation to create on target systems, right-click the folder that you want to contain the new folder, and then click **New Folder**.
5. In the **Source computer's folders** pane, navigate to the folder that contains the file that you want to add.
6. Drag the file from the **Source computer's files** pane to the **Destination computer's files** pane.

Tip ▪ *If you want to add an entire folder into a destination folder, drag the folder from the **Source computer's folders** pane to the **Destination computer's folders** pane.*

Source Path Variables

When you add a file from a predefined folder, the Link To column in the Destination computer's files pane displays a path variable, rather than an absolute path for most files. This provides more portability for your installation projects. If you move the project to another development machine, the path variables assure that you do not have to update the paths for your files.

Note ▪ *Files that have already been added to projects created with earlier versions of InstallShield display the absolute path in the Link To column. To display the path variable, you need to delete the file and re-add it to the project.*

The supported default path variables are in the following table:

Table 4-3 ▪ Supported Default Path Variables

Variable	Default Path
<CommonFilesFolder>	C:\Program Files\Common Files
<ISProductFolder>	C:\Program Files\InstallShield\2021
<ISProjectDataFolder>	C:\InstallShield 2021 Projects*Your Project Name* where *Your Project Name* is the name of your project—minus the file extension
<ISProjectFolder>	C:\InstallShield 2021 Projects
<ISRedistPlatformDependentExpressFolder>	C:\Program Files\InstallShield\2021\Redist\Language Independent\i386 Express
<ISRedistPlatformDependentFolder>	C:\Program Files\InstallShield\2021\Redist\Language Independent\i386
<ProgramFilesFolder>	C:\Program Files
<SystemFolder>	C:\WINDOWS\system32
<VSSolutionFolder>	Varies. This variable references a higher-level base directory. This support enables you to have in your InstallShield projects static links to files in sibling projects that are within the Visual Studio solution folder. To learn more, see Using the VSSolutionFolder Path Variable with Visual Studio Solutions.
<WindowsFolder>	C:\WINDOWS

Dragging and Dropping Files Using the Context Menu

The Files view enables you to drag and drop folders from your source computer to destinations on the target system. You have a number of options when you drag files from the **Source computer's folders** pane to the **Destination computer's folders** pane.

Tip ▪ *When you are using InstallShield on a 64-bit system and you want to add to your project a system file whose source location is the 64-bit System folder (System32) on your development machine, it is not possible to drag the file from one of the source computer panes at the top of the Files view to the appropriate location in one of the destination computer panes. To learn more, see Adding Files from Your 64-Bit Source Machine's 64-Bit System32 Folder.*

Task ***To display the context menu commands:***

1. In the View list under **Specify Application Data**, click **Files**.
2. In the **Source computer's folders** pane or the **Source computer's files** pane, right-click a folder or file and drag it to the **Destination computer's folders** pane or the **Destination computer's files** pane. Then release the mouse button.

   If the predefined folder to which you adding the folder or file is not displayed in the **Destination computer's folders** pane, you can add it. To learn how, see Displaying Predefined Folders in the Files View.

InstallShield displays a context menu that contains several commands.

Table 4-4 ▪ Commands Available from the Context Menu

Option	Description
Add	Adds the folder, subfolders, and/or files that are selected. This is the same as the default drag-and-drop behavior.
Add Preserving Source Structure	Adds the selected folder, subfolders, and/or files while preserving the file/folder structure found on the source computer. This command is available only if the source folder matches a predefined destination folder. In addition, the destination on which you drop the file/folder must be the Destination Computer in the **Destination computer's folders** pane.
Add Folder(s) Only	Adds only the selected folders and any subfolders that are in the selected folder. This option does not add any files that are within the selected folders or subfolders.
Add Folder(s) Only Preserving Source Structure	Adds only the selected folders and any subfolders that are in the selected folder. This option does not add any files that are within the selected folders or subfolders. This option also preserves the folder structure found on the source computer. This command is available only if the source folder matches a predefined destination folder. In addition, the destination on which you drop the file/folder must be the Destination Computer in the **Destination computer's folders** pane.
Cancel	Ends the drag-and-drop operation without making any changes.

Tip ▪ *If you are creating a 64-bit installation that can install to 64-bit locations (such as the Program Files Folder instead of the Program Files Folder (x86) folder) on 64-bit target systems, use a predefined 64-bit folder (ProgramFiles64Folder, CommonFiles64Folder, or System64Folder) or one of its subfolders in the* ***Destination computer's folders*** *pane.*

Note that a 64-bit installation cannot be run on 32-bit target systems. To learn more, see Challenges of Supporting Both 32-Bit and 64-Bit Target Operating Systems.

Adding Files from Your 64-Bit Source Machine's 64-Bit System32 Folder

On 64-bit systems, the System32 folder is reserved for 64-bit applications. When you try to view your development machine's 64-bit System folder from within the Files view in InstallShield, Windows redirects the view to instead display the SysWOW64 folder—the 32-bit version of the folder. Thus, when you are using InstallShield on a 64-bit system and you want to add to your project a system file whose source location is the 64-bit System folder (System32) on your development machine, it is not possible to drag the file from one of the source computer panes at the top of the Files view to the appropriate location in one of the destination computer panes.

To work around the redirection and add a 64-bit System32 file on your development machine to your InstallShield project, you can browse to the Sysnative folder on your machine and then select the appropriate file for your project. The following instructions explain how.

Caution ▪ *Note that in many cases, including system files in an installation is not recommended, since the system folder is protected by Windows. In these cases, the preferred way to deliver and update system files is to use a Microsoft redistributable, if one is available for the technology, or to have end users obtain the updates through Windows Update.*

Task ***To add to your project a file in the 64-bit System32 folder of a development system that has 64-bit Windows:***

1. In the View list under **Specify Application Data**, click **Files**.
2. In the **Destination computer's folders** pane, click the folder into which you want to place the file.
3. Right-click in the **Destination computer's folders** pane, and then click **Add File**. The **Open** dialog box opens.
4. Specify the following path (but replace the drive letter with the applicable drive letter if appropriate):

   ```
   C:\Windows\Sysnative
   ```

5. Select the appropriate file that you want to add to your project, and then click the **Open** button.

InstallShield adds the file to your project. InstallShield uses the Sysnative folder as part of the path for the source file that you added. WOW64 sees the Sysnative folder as a special alias; the file system does not redirect access away from this folder.

Availability of Sysnative Folder Support on Windows-Based Systems

Use of the Sysnative folder is not supported on 32-bit machines. If you use the Sysnative folder in an InstallShield project on a 64-bit system but then you try to build a release in that InstallShield project on a 32-bit system, InstallShield generates one or more build errors or warnings informing you that it could not find the source file.

Specifying Hard-Coded Destination Directories

Task ***To specify a particular drive as a hard-coded destination:***

1. In the View list under **Specify Application Data**, click **Files**.
2. In the **Destination computer's folders** pane, right-click **Destination Computer** and click **Add**. InstallShield adds a new folder to the **Destination Computer** explorer, and the folder name is selected for editing.
3. Type the drive letter followed by a colon (for example, `C:`).
4. Press ENTER.

Task ***To specify folders and subfolders beneath a drive letter folder to create a hard-coded destination path:***

1. Right-click the drive folder (for example, `C:`) under which you want to add a folder or the folder under which you want to add a subfolder and then click **Add**. InstallShield add a new folder, and the folder name is selected for editing.
2. Type the folder name.
3. Press ENTER.

Removing Files and Folders from Target Systems

InstallShield has built-in support that makes it easy to specify files and folders that you want to be removed from target systems at run time. This file and folder removal capability is useful for scenarios such as removing application-created files that your installation does not otherwise track.

You can schedule the removal of a file or folder for one of the following events:

- When the file or folder's feature is being installed
- When the file or folder's feature is being uninstalled
- When the file or folder's feature is being installed or uninstalled

Note that if the item to be removed is a folder, that folder is removed only if it is empty.

Task ***To configure file and folder removals:***

1. In the View list under **Specify Application Data**, click **Files**.
2. In the **Destination computer's folders** pane, select the destination folder that contains the file or folder that you want to be removed.
3. Right-click in the **Destination computer's files** pane and click **Add File Removal**. The **Properties** dialog box opens.
4. Configure the settings as needed. For details, see File Removal Properties Dialog Box.

InstallShield adds a file or folder icon to the **Destination computer's files** pane. The icon has a red X through it to indicate that it is referencing an item to be removed.

Tips for Managing Files and Folders in Your Project

Through drag-and-drop operations, common keyboard shortcuts such as CTRL+C and CTRL+P, and context menus that are available when you right-click items, you can easily move the files and folders to different destination folders and to different features.

Managing the Destination of Files and Folders in Your Project

Task ***To manage the destination of files and folders in your project:***

1. In the View list under **Specify Application Data**, click **Files**.
2. Do any of the following:
   - To change the settings for a file or folder, right-click the item and then click **Properties**. The **Properties** dialog box opens, enabling you to edit the settings as needed.
   - To move a file from one destination folder to another, drag it from one location in the **Destination computer's files** pane to the appropriate folder in the **Destination computer's folders** pane.
   - To move a folder from one destination folder to another, drag it from one location in the **Destination computer's folders** pane to the appropriate folder in the same pane.
   - To copy a file or folder from one location to another, press and hold CTRL while dragging the item from one location to another.

     As an alternative, you can copy an item to your Clipboard and then paste it into the appropriate location. Right-click the item and click **Copy**, or click it and press CTRL+C. Next, select another folder that you want to contain the file. Then right-click its folder and click **Paste**, or click the folder and press CTRL+P.
   - To delete a file or folder from your project, right-click it and then click **Delete**.

Managing the Feature Associations for Files in Your Project

Task ***To manage the feature associations of files in your project:***

1. In the View list under **Specify Application Data**, click **Files and Features**.
2. Select the feature that contains the file whose feature association you want to manage. InstallShield displays its files in the **Files** pane on the right.
3. Do any of the following:
   - To move a file from one feature to another, drag it from the **Files** pane and drop it onto the appropriate feature.
   - To copy a file from one feature to another, press and hold CTRL while dragging the file from one feature to another.

As an alternative, you can copy a file to your Clipboard and then paste it into the appropriate feature. Right-click the file and click **Copy**, or click it and press CTRL+C. Next, select another feature that you want to contain the file. Then right-click its Files pane, and click **Paste**, or click it and press CTRL+P.

- To remove a file from a feature, right-click the file and then click **Remove**.

Specifying Target System Requirements for Individual Files

In the Files view, you can specify target operating system requirements for each file.

Task ***To specify target operating system requirements:***

1. In the View list under **Specify Application Data**, click **Files**.
2. In the **Destination computer's files** pane, right-click a file and click **Properties**. The **Properties** dialog box opens.
3. Click the **Advanced** tab.
4. In the **Target operating system** area, indicate the operating systems for which this file is intended. To select a specific operating system, you need to clear the **All operating systems** check box.

Dynamic File Linking

If you want to add the contents of an entire directory to your project, you can do so through the use of dynamic file linking. When you select a source folder for dynamic linking, InstallShield adds the files within that folder to your release at build time. InstallShield scans the source folder before every build and automatically incorporates any new or changed files into your release. Dynamic file linking is useful when the list of files in a folder—and possibly the list of files in its subfolders—might change between builds.

Important ▪ *Dynamic file linking should be used with caution. If you inadvertently delete a dynamically linked file from the source folder that your dynamic link references, that file is not included in your release the next time you build it, and InstallShield does not display any build warning or error. Your product may install without any issues, but it may not work as expected, since the dynamically linked file that was inadvertently deleted is no longer being installed. Therefore, it is recommended that you avoid using dynamic file links for critical executable files—such as .exe, .dll, or .ocx files—especially if your product requires them in order to run successfully.*

Whenever possible, it is better to use the best practice method, instead of the by-directory method, for creating dynamic file links. Note that with both methods, however, a patch may not install correctly if a file that was present in a target image is removed from the dynamic link for the patch.

Filtering Dynamically Linked Files

When you are configuring your dynamic file link, you can specify whether you want to include subfolders of the dynamically linked folder. To further filter the files that are dynamically linked, you can identify specific names of files that you want to be included in or excluded from the dynamic link. In addition, you can use wild cards to specify only certain files or file types to be added or excluded.

For example, if all of your image files are in one folder along with sound files and you want to dynamically link only the image files, you could specify that you would like to include only .bmp and .ico files in the dynamically linked folder. To do so, you would use an asterisk (*) in your include pattern, as in the following example:

```
*.bmp, *.ico
```

To include or exclude a specific file, you would enter the full file name in the include or exclude pattern box. For more details, see Adding Files Dynamically.

Distinguishing Dynamically Linked Files and Folders from Static Files and Folders in the InstallShield Interface

When a dynamic file is displayed within the InstallShield interface, the lower-left corner of the file's icon includes an image that indicates that it is a dynamically linked file:

InstallShield includes that same dynamic file image on the icon of subfolders that are included in dynamic file links:

The icons that the InstallShield interface displays for static files and folders do not include this dynamic link image.

Limitations of Dynamic File Linking

Important ▪ *Dynamic file linking should be used with caution. If you inadvertently delete a dynamically linked file from the source folder that your dynamic link references, that file is not included in your release the next time you build it, and InstallShield does not display any build warning or error. Your product may install without any issues, but it may not work as expected, since the dynamically linked file that was inadvertently deleted is no longer being installed. Therefore, it is recommended that you avoid using dynamic file links for critical executable files—such as .exe, .dll, or .ocx files—especially if your product requires them in order to run successfully.*

Whenever possible, it is better to use the best practice method, instead of the by-directory method, for creating dynamic file links. Note that with both methods, however, a patch may not install correctly if a file that was present in a target image is removed from the dynamic link for the patch.

Note the following limitations if you are considering dynamic file linking in a project:

- You cannot create custom actions to a dynamically linked file.
- You cannot associate a file extension with a dynamically linked file.
- You cannot extract COM information from a dynamically linked file.
- You cannot set any properties such as Shared, Permanent, or Never Overwrite for a dynamically linked file.
- You cannot use the .NET installer class functionality for a dynamically linked file.
- You cannot specify that COM interop should be enabled for a dynamically linked file.
- You cannot change default file settings (such as Read-Only or Hidden).

- You cannot set file permissions for a dynamically linked file.
- You cannot create shortcuts to a dynamically linked file.
- You cannot perform a static or dynamic scan of a dynamically linked file.
- You cannot launch a dynamically linked file from the Setup Complete Success end-user dialog.

Any file that you add directly (not through dynamic linking) to your project has an internal name (FileKey). When you create a custom action, a file extension, shortcut, or other type of item, it actually points to this internal name.

When you add files to your project through dynamic links, the files are not physically added to the project. This means that these files do not have any FileKeys that can be associated with custom actions, file extensions, etc.

Determining the Appropriate Component Creation Method for Dynamically Linked Files

InstallShield provides two methods for creating components for dynamically linked files: the best practice method and the one-component-per-directory method. (A component is the smallest installable part of a product. The Express edition of InstallShield creates components for you automatically. For more information, see Installation Fundamentals.)

Using the Best Practice Method

When best practices for dynamic link creation are followed, InstallShield performs the following tasks at build time for all of the files that meet the include and exclude filter criteria of your dynamic link:

- InstallShield creates a separate component for each portable executable (PE) file in the dynamically linked folder. Each PE file is the key file of its component.
- InstallShield adds all non-PE files at the root level of the dynamic link to the component that contains the link.
- If the dynamic link includes a subfolder, InstallShield creates a new component for all of the non-PE files in that subfolder. If the dynamic link includes more than one subfolder, InstallShield creates a separate component for all of the non-PE files in each subfolder.

This is the default functionality for all new dynamic links.

***Tip** ▪ The File Extensions tab on the Options dialog box is where you specify which file types are PE files.*

Using the By-Directory Method

When the by-directory method is used for dynamic link creation, InstallShield performs the following tasks at build time for all of the files that meet the include and exclude filter criteria of your dynamic link:

- InstallShield creates one component for all of the files that are in the root-level dynamically linked folder, regardless of the file types.
- If the dynamic link includes one or more subfolders, InstallShield creates a separate component for all of the files in each subfolder, regardless of the file types. The first dynamically linked file in a subfolder's component is the key file of that component.

This method of dynamic link creation is the traditional method that was available in InstallShield before the best practice method was introduced.

Determining Which Dynamic Link Creation Method to Use

For most dynamic links, the preferred dynamic link creation method is the best practice method. This method, in combination with the streamlined QuickPatch package functionality, enables you to create patches according to Windows Installer component rules.

The best practice method is also preferred if it is possible that you may someday upgrade your project to the Premier or Professional editions of InstallShield. These editions let you create minor upgrades and small updates. For minor upgrades and small updates, the components, key files, and feature-component organization need to be maintained across the earlier and later .msi databases; for patches, the `File` table keys also need to be maintained. Since each component name and component code—and possibly key files—change at each build with the by-directory method of dynamic file linking, issues may occur. The advantage of the best practice method is that it allows for greater predictability than with the by-directory method.

Note that if you want to create a patch for an earlier version of your product, and the earlier installation includes dynamic links that used the by-directory method, you must continue to use the by-directory method for the same dynamic links. However, if you add new dynamic links in your upgrade project, you can use the best practice method for those new dynamic links. That is, you can mix both types of dynamic linking in the same project and create a patch to deliver the upgrade.

Important ▪ *Whenever possible, it is better to use the best practice method, instead of the by-directory method, for creating components for dynamically linked files. Note that with both methods, however, a minor upgrade, a small update, or a patch may not install correctly if a file that was present in a target image is removed from the dynamic link for the upgrade or patch.*

Note ▪ *For information on the rules that Windows Installer uses when determining whether a file included in a package should overwrite a file that already exists on the target system, see* *Overwriting Files on the Target Machine**.*

Specifying Which Dynamic Link Creation Method You Want to Use

The File Linking tab on the Folder Properties dialog box is where you specify which component creation method you want to use.

Adding Files Dynamically

Task ***To dynamically add files to a folder in your installation:***

1. In the View List under **Specify Application Data**, click **Files**.
2. In the **Feature** list, select the feature with which you want the files associated.
3. In the **Destination computer's folders** pane, right-click the folder that should contain the dynamically linked files, and then click **Dynamic File Linking**. The **Properties** dialog box opens.

Tip ▪ *If the folder that should contain the dynamically linked files does not already exist in the* ***Destination computer's folders*** *pane, you must first create it. To learn how, see Adding Files and Folders to a Project.*

4. Click the File Linking tab.
5. Define the dynamic link and click **OK**.

Overwriting Files on the Target Machine

In the Files view, you can specify the overwrite property of files in your installation. The Windows Installer service uses the overwrite property of each file to determine whether it should be replaced if a file in the installation already exists on the target system.

Table 4-5 ▪ File Overwrite Rules

File Overwrite Property Value	Behavior on the Target Machine
Windows Installer Versioning Rules (Recommended)	If you select this option, Windows Installer versioning rules are used to determine whether a file that already exists on the target system should be replaced. Windows Installer enforces the following rules: • **Versioned files**—In all cases, the file with the highest version is maintained, even if the file already on the target machine has a higher version than the one being installed. Additionally, a file of any version is maintained over unversioned files. • **File language**—All other things being equal, the file that is the same language as the installation is maintained over different language versions of the file. The only exception to this rule applies to multiple language files. Files with multiple languages are maintained over single language versions of a file. • **Date**—If the modified date of a file already present on the target machine is later than the creation date of that file, the file is not overwritten. This rule protects user preference files from being wiped out during an upgrade or reinstallation.
Never Overwrite	If you select this option, the file—if it exists on the target system—is never overwritten, regardless of the file version.
Always Overwrite	If you select this option, the file—if it exists on the target system—is always overwritten, regardless of the file version.

Displaying Predefined Folders in the Files View

Task ***To display a predefined destination folder:***

1. In the View list under **Specify Application Data**, click **Files**.
2. In the **Destination computer's folders** pane, right-click **Destination Computer**, point to **Show Predefined Folders**, and then click the type of predefined folder that you want to use.

Finding Files and Folders in Your Project

If you have added numerous folders and files to your project, you may have trouble finding a particular folder or file. You can perform a search for folders and files in the Files view; InstallShield locates any matches and highlights the first one. You can keep searching until you find all matches for your search criteria.

Task ***To find files and folders in your project:***

1. In the View list under **Specify Application Data**, click **Files**.
2. In the **Destination computer's folders** pane, select **Destination Computer**.
3. On the **Edit** menu, click **Find**. The **Find** dialog box opens.

   As an alternative, you can press CTRL+F.
4. In the **Find What** box, type the text to be found. You can use wildcard expressions, such as **`*.exe`**.
5. In the **Look at** area, specify whether you want to search for files, folders, or both.
6. Specify any other desired criteria.
7. Click **Find Next**. The first item (if any) that matches your search criteria is selected in either the **Destination computer's folders** pane or the **Destination computer's files** pane.
8. To find the next item (if any) that matches your criteria, press the F3 key. Repeat this step as necessary.

Configuring Permissions for Files and Folders

InstallShield lets you configure settings for securing files and folders for end users who run your product in a locked-down environment. You can assign permissions for a file or folder to specific groups and users. For example, you may assign Read, Write, and Delete permissions for a particular file to the Administrators group, but only Read permissions for all of the users in a different group.

Task ***To configure the permissions for a file or folder:***

1. In the View List under **Specify Application Data**, click **Files**.
2. For a file: In the **Destination computer's files** pane, right-click the file and then click **Properties**. The **Properties** dialog box opens.

For a folder: In the **Destination computer's folders** pane, right-click the folder and then click **Properties**. The **Properties** dialog box opens.

3. Click the **Permissions** button. The **Permissions** dialog box opens.
4. Add, modify, and remove permissions entries as needed. For more information, see Permissions Dialog Boxes for Files and Directories.

Depending on what is selected for the Locked-Down Permissions setting in the General Information view of your project, InstallShield adds permissions data to either the `ISLockPermissions` table or the `LockPermissions` table. To learn more, see Securing Files, Folders, and Registry Keys in a Locked-Down Environment.

Including Redistributables in Your Installation

InstallShield includes many commonly used third-party redistributables, making it easy to add support for popular technologies such as the .NET Framework to your installation. When you add redistributables to your project, the redistributables, plus all of the associated dependencies, are added to your installation. This simplifies the process of packaging redistributables and ensures consistency for internal or external use.

The Redistributables view contains all of the InstallShield prerequisites, merge modules, and objects that are included with InstallShield.

InstallShield Prerequisites

An InstallShield prerequisite is an installation for a product or technology framework that is required by your product. You can add any of the existing InstallShield prerequisites to your installation projects.

Including InstallShield prerequisites in your project enables you to chain multiple installations together, bypassing the Windows Installer limitation that permits only one Execute sequence to be run at a time. The `Setup.exe` setup launcher serves as a bootstrap application that manages the chaining.

***Edition** ▪ InstallShield Premier Edition and InstallShield Professional Edition include the InstallShield Prerequisite Editor, a tool that enables you to define custom InstallShield prerequisites and to edit settings for any existing InstallShield prerequisites.*

InstallShield includes support for two types of InstallShield prerequisites:

- **Setup prerequisite**—The installation for this type of prerequisite runs before your installation runs.
- **Feature prerequisite**—This type of prerequisite is associated with one or more features. It is installed if the feature that contains the prerequisite is installed and if the prerequisite is not already installed on the system. Thus, if a feature has a condition that is not met on the target system, or if the end user chooses not to install the feature, the feature is not installed. As a result, none of its associated feature prerequisites are installed, unless the feature prerequisites are also associated with other features that are installed.

Merge Modules

A merge module (or .msm file) contains all of the logic and files needed to install distinct pieces of functionality. For example, some applications require Microsoft C++ run-time libraries. Instead of having to include the file in a feature and figure out its installation requirements, you can simply attach the Microsoft C++ runtime library merge module to one of your project's features.

Note ▪ *Many of the merge modules included in the Redistributables view are authored by Microsoft or another third party. InstallShield distributes these modules as a courtesy to assist you in creating your installation project. However, InstallShield cannot modify or fix any problems that may exist within third party–authored modules. You are encouraged to contact the vendor regarding issues with specific third party–authored modules.*

Objects

Like merge modules, objects contain logic and files needed to install distinct pieces of functionality. Some objects, such as the DirectX object included with InstallShield, require customization through a wizard. As soon as you add such an object to your installation, its customization wizard opens. You can either customize your object at the time you add it, or cancel the wizard and customize your object later by right-clicking the object and selecting Change Objects Settings.

Redistributables Gallery

Because the file size of many of the redistributables is so large, some that are available for use in your projects are not added to your computer when you install InstallShield. However, these redistributables are still available for download from the Internet to your computer.

Configurable Merge Modules

Shipping Redistributable Files

InstallShield provides third-party redistributables that you can incorporate into your installation projects. If you include redistributable technology in your projects—for example, Crystal Reports—that redistributable must be licensed from the vendor. You cannot legally redistribute these technologies without the appropriate licensing. For details, consult the vendor's documentation.

When you build releases in an InstallShield project, InstallShield includes various InstallShield redistributable files in the build output. You may use these InstallShield redistributable files in the build output according to the InstallShield End-User License Agreement. Most of these files are installed in the *InstallShield Program Files Folder*\Redist folder and included in builds as needed. Following is a list of the InstallShield redistributable files.

- _isres_LanguageID.dll
- ClrSuitePSHelper.dll
- ClrWrap.dll
- CommonHelper.dll
- corecomp.ini
- default.pal
- DLLWrap.dll
- dotnetfx.exe
- DotNetInstaller.exe
- EulaScrollWatcher.dll

- FileBrowse.dll
- IISHelper.dll
- IISRT.dll
- InstallShield.ClrHelper.dll
- InstallShield.Interop.Msi.dll
- ISBEW64.exe
- ISChain.exe
- ISChainPackages.dll
- ISComSrv.dll
- ISExpHlp.dll
- isexternalui.dll
- IsLockPermissions.dll
- ISNetAPI.dll
- ISNetApiRT.dll
- ISNetworkShares.dll
- ISRegSvr.dll
- Isrt.dll
- ISScheduledTasks.dll
- IsSchRpl.dll
- ISSetup.dll
- ISSQLSrv.dll
- IsWebDeploy.dll
- ISWindowsFeaturesAction.dll
- ISWindowsFeaturesAction64.dll
- ISXmlCfg.dll
- Layout.bin
- PowerShellWrap.dll
- PrqLaunch.dll
- QuickPatchHelper.dll
- SerialNumCAHelper.dll
- SetAllUsers.dll
- Setup.exe
- Setup.ini

- setup.inx
- setup.isn
- setup.ocx
- setup.skin
- Setup_UI.dll
- setupPreReq.exe
- SetupSuite.exe
- SetupSuite64.exe
- SFHelper.dll
- SQLRT.dll
- SuiteAppxHelper.exe
- XMLRT.dll
- Image files that are installed in the subfolders in the following folder:

  InstallShield Program Files Folder\Support\Themes

- Image and icon files that are installed in the following folder, as well in this folder's scale-150 and scale-200 subfolders:

  InstallShield Program Files Folder\Redist\Language Independent\OS Independent

Managing the Redistributables Gallery

Because the file size of many of the redistributables is so large, some that are available for use in your projects are not added to your computer when you install InstallShield. However, these redistributables are still available for download from the Internet to your computer.

You can identify the status of a redistributable by its icon. Following is a list of the possible icons and a description of each:

Table 4-6 ▪ Redistributable Icons

Icon	Description
	This InstallShield prerequisite is installed on your computer.
	This InstallShield prerequisite is not installed on your computer but it is available for download.
	This InstallShield prerequisite is included in your project but its location is not listed in one of the directories that is specified on the Prerequisites tab of the Options dialog box.
	This merge module is installed on your computer.

Table 4-6 ▪ Redistributable Icons (cont.)

Icon	Description
	This merge module is not installed on your computer but it is available for download.
	An old version of this merge module is installed on your computer. A new version is available for download.
	This object is installed on your computer.
	This object is not installed on your computer but it is available for download.
	An old version of this object is installed on your computer. A new version is available for download.

Tip ▪ *If you add to your installation project an object or merge module that is not installed on your computer, a build error is generated when you build your project. To eliminate the build error, either remove the redistributable from your project or download it before rebuilding your release. If a redistributable is not installed on your computer,* ***Needs to be downloaded*** *is specified in the Location column for that redistributable.*

Working with the Redistributables Gallery

The Redistributables view displays the redistributables gallery, which consists of InstallShield prerequisites, merge modules, and objects, that you can include with your installations.

InstallShield Prerequisites

Many InstallShield prerequisites are available in InstallShield. All InstallShield prerequisite (.prq) files are stored in the following location:

InstallShield Program Files Folder\SetupPrerequisites

Merge Modules

Merge modules are available from a variety of sources. Although InstallShield includes many redistributable modules, new versions may be available or other software developers may have released a module that you need.

The source of the merge module files listed in the Redistributables view is the folder or folders specified on the Merge Modules tab of the Options dialog box. To access the Options dialog box, on the Tools menu, click Options.

The following directory is the default location for the modules that come with InstallShield:

InstallShield Program Files Folder\Modules\i386

Objects

InstallShield provides many redistributable objects. Furthermore, you may want to add to your projects the objects that other developers created.

The default location for the objects that come with InstallShield is:

InstallShield Program Files Folder\Objects

The objects that are included in the above location are listed in the Redistributables view.

Downloading Redistributables to Your Computer

The Redistributables view enables you to download the latest InstallShield prerequisites, merge modules, and objects from the Revenera Web site to your computer. If a redistributable is not installed on your computer, **Needs to be downloaded** is specified in the Location column for that redistributable.

Task ***To download a specific InstallShield prerequisite, merge module, or object:***

1. In the View List under **Specify Application Data**, click **Redistributables**.
2. To specify which types of redistributables—all types, InstallShield prerequisites, merge modules, or objects—should be displayed, select the appropriate option in the **Object types to display** list.
3. Right-click the InstallShield prerequisite, merge module, or object that you would like to download and then click **Download Selected Item**.

Task ***To download all of the InstallShield prerequisites, merge modules, and objects that are needed for your installation project:***

1. In the View List under **Specify Application Data**, click **Redistributables**.
2. Right-click any InstallShield prerequisite, merge module, or object and then click **Download All Required Items**.

Adding InstallShield Prerequisites to the Redistributables Gallery

Task ***To add an InstallShield prerequisite to the redistributables gallery:***

1. Acquire the new or updated InstallShield prerequisite (.prq) file.
2. Using Windows Explorer, copy the new prerequisite to the following location:

   InstallShield Program Files Folder\SetupPrerequisites
3. Close InstallShield if it is currently open.
4. Launch InstallShield.

The modifications made are reflected in the Redistributables view.

Removing InstallShield Prerequisites from the Redistributables Gallery

Task ***To remove an InstallShield prerequisite from the redistributables gallery:***

1. Close InstallShield.
2. Using Windows Explorer, locate and delete the InstallShield prerequisite that you want to remove from the gallery. InstallShield prerequisites are located in the following directory:

   InstallShield Program Files Folder\SetupPrerequisites
3. Launch InstallShield.

The modifications that you made are reflected in the Redistributables view.

Browsing for Merge Modules

If a merge module that you would like to add to your project is not listed in the Redistributables view, you can browse to find it and also add it to your project and this view.

Task ***To browse to a merge module:***

1. In the View List under **Specify Application Data**, click **Redistributables**.
2. Right-click an item and click **Browse for Merge Module**. The **Open** dialog box opens.
3. Browse to the merge module file.
4. Click **OK**.

What Happens When You Browse for a Merge Module

InstallShield does not maintain references to merge modules as explicit paths. Instead, it generates a key for a merge module based on the merge module GUID and the merge module locale. When InstallShield needs to access the merge module, it looks in the folders specified in the Merge Module Locations box for a file that matches that key. The Merge Module Locations box is on the Merge Modules tab of the Options dialog box.

When you browse for a merge module, the path to the folder containing the merge module is added to the list of paths in the Merge Module Locations box. In addition, a GUID:Locale key is added to your installation project based on the selected file.

Impact on Your Installation

If two merge modules in the Merge Module Locations box have the GUID:Locale key, only one is included into your installation, even if they have different file names. Because of the way InstallShield uses the Merge Module Locations box to search, it is not possible to predict which merge module will be included.

Limiting the Number of Directories in the Merge Module Locations Box

If you use a shared merge module gallery, there might be earlier or later versions of a merge module in the gallery than what exists on the target machine. For this reason, it is sometimes prudent to try and limit the number of directories in your Merge Module Locations box.

Task ***To limit the number of directories, do one of the following:***

- Using Windows Explorer, copy the merge module that you want into one of the folders that is already listed in the **Merge Module Locations** box.
- Remove the default folders from the search path so that you are referencing only the shared location.

Adding Merge Modules to the Redistributables Gallery

Task ***To add a merge module to the redistributables gallery:***

1. Acquire the new or updated merge module.
2. Using Windows Explorer, copy the new module to one of the folders specified on the Merge Module Options tab of the **Options** dialog box.

   The default location for the modules that come with InstallShield is:

   `Program Files Folder\`*`InstallShield Folder`*`\Modules\i386`
3. Close InstallShield if it is currently open.
4. Launch InstallShield.

The modifications that you made are reflected in the Redistributables view.

Removing Merge Modules from the Redistributables Gallery

Task ***To remove a merge module from the redistributables gallery:***

1. Close InstallShield if it is currently open.
2. Using Windows Explorer, locate and delete the merge module that you want to remove from the gallery. Ensure that you search each directory specified on the Merge Module Options tab of the Options dialog box.
3. Launch InstallShield.

The modifications that you made are reflected in the Redistributables view.

Note ▪ *If you delete a merge module that is currently associated with your installation, the message* ***[Merge Module Not Found]*** *is displayed to inform you that the module cannot be included in your installation.*

Incorporating InstallShield Prerequisites, Merge Modules, and Objects in Projects

InstallShield includes many third-party redistributables that are packaged as InstallShield prerequisites, merge modules, and objects. You can add these built-in redistributables to your installation projects. To learn how, see this section of the documentation.

***Edition** ▪ InstallShield Premier Edition and InstallShield Professional Edition include the InstallShield Prerequisite Editor, a tool that enables you to define custom InstallShield prerequisites and to edit settings for any existing InstallShield prerequisites. In addition, these editions include a merge module project type, which you can use to create your own merge modules.*

Adding InstallShield Prerequisites, Merge Modules, and Objects to Projects

Two types of redistributables—merge modules and objects—must be associated with a feature in order to be installed. You can associate a single merge module or object with as many features or subfeatures as needed.

When you add an InstallShield prerequisite to a project, it is not associated with any feature by default, and it is called a *setup prerequisite*, since it is run before your main installation runs. If appropriate, you can associate an InstallShield prerequisite with one more features that are currently in your project.

Task ***To add an InstallShield prerequisite, merge module, or object to a project:***

1. In the View List under **Specify Application Data**, click **Redistributables**.
2. Select the check box in front of the redistributable that you want to add. If you select an object, the associated wizard opens to guide you through the customization process.
3. For a merge module or an object: In the **Conditional Installation** pane, select the check box for each feature that should contain this redistributable.

   If you want to associate a prerequisite with a feature: In the **Conditional Installation** pane, select the check box for each feature that should contain this prerequisite. If you do not want to associate the prerequisite with a feature, leave the **Install before feature selection** check box selected. This check box is selected by default when you add an InstallShield prerequisite to a project.

***Tip** ▪ The right pane in the Redistributables view shows details about the merge module, object, or InstallShield prerequisite that is selected in the list of available redistributables. Review this details pane to find out information such as which files a redistributable installs. You can hide or show the details pane by clicking the Show Details button in this view.*

***Note** ▪ If **Needs to be downloaded** is specified in the Location column for the InstallShield prerequisite that you added to your project, that prerequisite is not installed on your computer. You can download the prerequisite from the Internet to your computer if you would like to include it in your project. If you build a release without first downloading one or more required prerequisites, and if you specify that the prerequisites should be extracted from `Setup.exe` or copied from the source media (instead of being downloaded from the Web to the end user's*

computer), one or more build errors may be generated. To eliminate the build errors, remove the prerequisite from your project, download it to your computer, or change the InstallShield prerequisites location for the release to the download option; then rebuild the release.

Obtaining InstallShield Prerequisites and Objects

Note that some of the InstallShield prerequisites and objects are not installed with InstallShield. You may need to download them. For more information, see Obtaining Updates for InstallShield.

Removing InstallShield Prerequisites, Merge Modules, or Objects from a Project

Task ***To remove an InstallShield prerequisite, merge module, or object from your project:***

1. In the View List under **Specify Application Data**, click **Redistributables**.
2. Clear the check box in front of the InstallShield prerequisite, merge module, or object that you want to remove from your installation.

InstallShield removes the redistributable and any dependencies associated with it automatically.

Determining the Files in InstallShield Prerequisites, Merge Modules, and Objects

If you need to see a list of files in an InstallShield prerequisite, a merge module, or an object, you can do so from within the Redistributables view. The right pane in this view shows details about the InstallShield prerequisite, merge module, or object that is selected in the list of available redistributables. This details pane provides information such as which files a redistributable installs. You can hide or show the details pane by clicking the Show Details button in this view.

Tip ▪ *For another way to see the files contained in a merge module or object, see Knowledge Base article Q106474. This article contains a link to a downloadable Merge Module Dependency Viewer.*

Working with InstallShield Prerequisites that Are Included in Projects

An InstallShield prerequisite is an installation for a product or technology framework that is required by your product. Some examples of InstallShield prerequisites that are included with InstallShield are Java Runtime Environment (JRE) and SQL Server Express Edition. You can add any of the existing InstallShield prerequisites to your installation projects.

Edition ▪ *InstallShield Premier Edition and InstallShield Professional Edition include the InstallShield Prerequisite Editor, a tool that enables you to define custom InstallShield prerequisites and to edit settings for any existing InstallShield prerequisites.*

Including InstallShield prerequisites in your project enables you to chain multiple installations together, bypassing the Windows Installer limitation that permits only one Execute sequence to be run at a time. The `Setup.exe` setup launcher serves as a bootstrap application that manages the chaining.

The Redistributables view is where you add InstallShield prerequisites to Express projects.

Setup Prerequisites vs. Feature Prerequisites

An InstallShield prerequisite that is run before the main installation's user interface sequence begins is called a *setup prerequisite*. Setup prerequisites are useful for base applications and technology frameworks that must be installed for all configurations of the installed product or that provide functionality that is used during the installation itself. When you add an InstallShield prerequisite to a project, it is the setup prerequisite type of InstallShield prerequisite by default.

The Express project type enables you to associate InstallShield prerequisites with features in your main installation. When an InstallShield prerequisite is associated with one or more features, it is called a *feature prerequisite*. Feature prerequisites are installed after an end user has chosen which features to install; like merge modules, a feature prerequisite is installed only if one or more of the features that contain it are installed. Thus, feature prerequisites are useful for applications or components that are used by only some configurations of the installed product and are not used during the installation itself.

Review the following sections for more information that will help you determine which type of InstallShield prerequisite will best fit your requirements.

Special Considerations for Setup Prerequisites

Following are some tips to consider if you are including one or more setup prerequisites in your project.

.NET Framework Requirements

If your product requires that the .NET Framework be installed on the target system, you may include the .NET Framework redistributable to your project. If the target system does not have the .NET Framework, it is installed during your installation. For more details, see Adding .NET Framework Redistributables to Projects.

Edition ▪ *InstallShield Premier Edition and InstallShield Professional Edition enable you to configure an InstallShield prerequisite so that it is installed either before or after any installation of the Windows Installer engine and the .NET Framework.*

Launching the .msi Package Instead of the Setup Launcher

If your installation includes a setup prerequisite and end users launch the .msi package for your product directly, rather than launch the `Setup.exe` setup launcher, the setup prerequisite installation will not run. If the prerequisite is not already present on a target system, your product may not work as expected. This scenario may occur if you build an uncompressed release, where the .msi package is not streamed into the `Setup.exe` file.

Special Considerations for Feature Prerequisites

Following are some tips to consider if you are including one or more feature prerequisites in your project.

Windows Installer Requirements

If your project includes a prerequisite that installs the Windows Installer, the prerequisite should be a setup prerequisite, not a feature prerequisite. That is, this prerequisite should not be associated with a feature.

.NET Framework Requirements

If your project include a prerequisite that installs the .NET Framework and your installation requires that the .NET Framework be present—for example, if your installation installs files to the GAC—the .NET Framework prerequisite should be a setup prerequisite, not a feature prerequisite. That is, this prerequisite should not be associated with a feature.

Potential Restart Issues for Feature Prerequisites

If you add an InstallShield prerequisite to your project and it may require a restart, it is recommended that you avoid associating the prerequisite with a feature. If a feature prerequisite does trigger a restart, the ReadyToInstall dialog is displayed again after the restart, and the end user will need to click the Install button again to proceed with the rest of the installation.

Calculations for Disk Space Requirements

When the Windows Installer performs the file costing–related actions, it does not automatically include the disk space that is required by any feature prerequisites. Therefore, if the CustomSetup dialog is displayed at run time, the disk space amounts that are listed for various features may be inaccurate, since they will not account for the disk space that is required by feature prerequisites. In addition, it is possible that a target system may have sufficient disk space for the main installation, but not for the feature prerequisites. In this scenario, the target system may run out of disk space midway through the installation.

Associating an InstallShield Prerequisite with a Feature in a Project

If an InstallShield prerequisite is associated with a feature in a project, it is considered to be a *feature prerequisite*. If it is not associated with a feature, it is considered to be a *setup prerequisite*.

Whenever you add an InstallShield prerequisite to an installation project, it is automatically added as a setup prerequisite by default. You can make it a feature prerequisite by associating it with one or more features that already exist in your project.

Task ***To associate an InstallShield prerequisite with a feature:***

1. In the View List under **Specify Application Data**, click **Redistributables**.
2. In the list of redistributables, select the InstallShield prerequisite that you want to associate with a feature.

   Note ▪ *The InstallShield prerequisite's check box must already be selected; this indicates that it is being included in your project. For more information, see Adding InstallShield Prerequisites, Merge Modules, and Objects to Projects.*
3. In the **Conditional Installation** pane, select the check box of each feature to which you want to add this InstallShield prerequisite.

If you want to associate the prerequisite with a new feature, you must first create it. To learn how to create a new feature, see Creating Features.

If you associate a prerequisite with all of the features in your project and then you later add a new feature, InstallShield does not automatically associate the feature prerequisite with the new feature.

Note ▪ *Feature prerequisites have some limitations that setup prerequisites do not have. For more information, see Setup Prerequisites vs. Feature Prerequisites.*

Disassociating an InstallShield Prerequisite from a Feature in a Project

If an InstallShield prerequisite is associated with a feature in a project, it is considered to be a *feature prerequisite*. If it is not associated with a feature, it is considered to be a *setup prerequisite*.

Task ***To remove an InstallShield prerequisite from a feature:***

1. In the View List under **Specify Application Data**, click **Redistributables**.
2. In the list of redistributables, select the InstallShield prerequisite that you want to disassociate from a feature.
3. In the **Conditional Installation** pane, select the **Install before feature selection** check box. This check box is selected by default when you add an InstallShield prerequisite to a project.

Note ▪ *Setup prerequisites have some advantages over feature prerequisites. For more information, see Setup Prerequisites vs. Feature Prerequisites.*

Specifying the Installation Order of InstallShield Prerequisites

The Redistributables view is where you specify the order in which InstallShield prerequisites should be installed if you include more than one in your project.

Task ***To specify the order in which the InstallShield prerequisites should be installed on the target machine:***

1. In the View List under **Specify Application Data**, click **Redistributables**.
2. Add the necessary InstallShield prerequisites to your project if you have not already done so.
3. Right-click any redistributable and click **Set InstallShield Prerequisite Order**. The **InstallShield Prerequisite Installation Order** dialog box opens.
4. Select a prerequisite in the list and then click the up or down arrow to move it up or down in the order for installation.

Note ▪ *Note that when you are specifying the order, InstallShield does not distinguish between setup prerequisites and feature prerequisites. Thus, if your project contains a mix of setup prerequisites and feature prerequisites, they are all listed in one combined list on the InstallShield Prerequisite Installation Order dialog box. At run time, before the main installation launches, the* `Setup.exe` *setup launcher evaluates only the setup prerequisites and—if appropriate—installs them in the order that you specified on the InstallShield Prerequisite*

Installation Order dialog box. Then later during the installation, the Windows Installer engine evaluates only the feature prerequisites and—if appropriate—installs them in the order that you specified.

To learn about the differences between setup prerequisites and feature prerequisites (which are the two types of InstallShield prerequisites), see Setup Prerequisites vs. Feature Prerequisites.

Configuring a Release that Includes InstallShield Prerequisites

When you package an installation that includes InstallShield prerequisites, you can use any one of the following methods for supplying the InstallShield prerequisite files to end users:

- Store the InstallShield prerequisite files on the source media.
- Compress the InstallShield prerequisite files into `Setup.exe`, to be extracted at run time, as needed.
- If necessary, your installation can download the InstallShield prerequisite files that are included in your project from the URL that is specified in the InstallShield prerequisite file (.prq) for each prerequisite.

You can specify different methods for each InstallShield prerequisite in your project. To learn more, see Specifying a Run-Time Location for a Specific InstallShield Prerequisite.

You can also override individual methods at the release level if you want all of the InstallShield prerequisites in a release to be available through the same method. For more information, see Specifying the Run-Time Location for InstallShield Prerequisites at the Release Level.

***Edition** ▪ InstallShield Premier Edition and InstallShield Professional Edition enable you to configure an InstallShield prerequisite so that it is installed either before or after any installation of the Windows Installer engine and the .NET Framework.*

Specifying the Directories that Contain InstallShield Prerequisites

The default location for InstallShield prerequisite files (.prq) is:

`InstallShield Program Files Folder``\SetupPrerequisites`

InstallShield lets you specify additional or alternative locations on your local machine, or on a network. This flexibility enables you to store InstallShield prerequisites in source code control and to share a common set of InstallShield prerequisites with other team members.

InstallShield offers several ways for specifying the search paths for InstallShield prerequisite files (.prq):

- If you are editing or building from within InstallShield, use the Prerequisites tab on the Options dialog box—which is displayed when you click Options on the Tools menu—to specify a comma-delimited list of machine-wide folders and current-user folders.
- If you are building from the command line with ISCmdBld.exe, use the `-prqpath` parameter to specify a comma-delimited list of folders.

  If you use an .ini file to specify ISCmdBld.exe parameters, you can use the PrerequisitePath parameter in the [Mode] section of your .ini file to specify a comma-delimited list of folders.

- If you are building through MSBuild or Team Foundation Server (TFS), use the PrerequisitePath parameter on the InstallShield task. This parameter is exposed as the ItemGroup InstallShieldPrerequisitePath when the default targets file is used. To specify multiple paths, use an ordered array of paths.

Instead of using hard-coded paths, you can use path variables in paths, as in the following example:

```
<ISProductFolder>\SetupPrerequisites,<ISProjectFolder>\MyCustomPrerequisites
```

The Redistributables view lists the names of the InstallShield prerequisites that correspond with the .prq files that are present in the various search paths that are specified on the Prerequisites tab of the Options dialog box. If the same .prq file is in multiple search paths, InstallShield shows only the first instance that it encounters. InstallShield first searches each path that is listed in the per-user setting on the Prerequisites tab. Then, InstallShield checks each path that is listed in the machine-wide setting.

At build time, if your project includes one or more InstallShield prerequisites, InstallShield searches the specified locations and includes the appropriate InstallShield prerequisites in your release as needed. If the same .prq file is in multiple search paths, InstallShield includes in the build only the first instance that it encounters. It uses the following order to search for .prq files:

1. InstallShield checks the paths that are specified through the `-prqpath` command-line parameter, the PrerequisitePath .ini file parameter, or the PrerequisitePath parameter on the InstallShield task.
2. InstallShield checks each path that is listed in the per-user setting on the Prerequisites tab.
3. InstallShield checks each path that is listed in the machine-wide setting on the Prerequisites tab.
4. If no paths are specified in any of the aforementioned locations, InstallShield checks the default location (*InstallShield Program Files Folder*\SetupPrerequisites).

Specifying a Run-Time Location for a Specific InstallShield Prerequisite

InstallShield enables you to specify a different run-time location for each InstallShield prerequisite in your project.

Task ***To specify a different location for each InstallShield prerequisite in your installation:***

1. In the View List under **Specify Application Data**, click **Redistributables**.
2. Select the check box for one of the InstallShield prerequisites that you want to include in your installation.
3. Right-click the InstallShield prerequisite and click **Properties**. The **InstallShield Prerequisites Properties** dialog box opens.
4. In the **Build Location** list, click the appropriate option.

Note that the location that you specify can be overridden at the release level. To avoid overriding the value that you selected for an individual InstallShield prerequisite, the InstallShield Prerequisites Location setting at the release level setting must be set to Follow Individual Selections. For more information, see Specifying the Run-Time Location for InstallShield Prerequisites at the Release Level.

Building a Release that Includes InstallShield Prerequisites

When InstallShield builds a `Setup.exe` file for a project that does not include any prerequisites, it starts with the base `Setup.exe` file stored in the following location:

InstallShield Program Files Folder\redist\Language Independent\i386

However, when InstallShield builds a Setup.exe file for a project that includes prerequisites, the aforementioned Setup.exe file cannot be used as the base because it does not have the capability of including prerequisites. A slightly larger file called SetupPrereq.exe is used instead. This base SetupPrereq.exe file is located in the same directory as the base Setup.exe file. Since two different base files—Setup.exe and SetupPrereq.exe—are used, only installation authors who are actually including prerequisites in their projects incur the additional size overhead in the final, built Setup.exe file that is distributed to end users.

Run-Time Behavior for an Installation that Includes InstallShield Prerequisites

Tip ▪ *To learn about the differences between setup prerequisites and feature prerequisites (which are the two types of InstallShield prerequisites), see Setup Prerequisites vs. Feature Prerequisites.*

Overview of an Installation that Includes InstallShield Prerequisites

The following procedure explains what typically occurs at run time when an end user launches an installation that includes setup and feature prerequisites.

1. The setup launcher (typically called Setup.exe) displays the language selection dialog if appropriate.
2. The setup launcher displays the setup prerequisite dialog and launches the setup prerequisite installations if appropriate.
3. The installation displays the installation UI, which may allow the end user to select features or configure items. The installation UI shows a progress dialog.
4. The setup launcher launches the feature prerequisite installations if appropriate:
    a. The built-in InstallShield custom action ISInstallPrerequisites, which is scheduled between the SetupProgress dialog and the ExecuteAction action, compares the features that were selected for installation against the list in the Windows Installer property IsPrerequisiteFeatures. If there are no matches, no feature prerequisites are installed.
    b. The ISInstallPrerequisites action attempts to find and launch the setup launcher, and it provides the list of features that are being installed. The path to the setup launcher is identified by the Windows Installer properties SETUPEXEDIR and SETUPEXENAME:

        [SETUPEXEDIR]\[SETUPEXENAME]

        If ISInstallPrerequisites cannot find the setup launcher in that location, it searches elsewhere. For a first-time installation, ISInstallPrerequisites checks SourceDir. For maintenance mode, ISInstallPrerequisites checks paths that are related to the installation source path.

        If ISInstallPrerequisites still cannot find the setup launcher, or if it finds multiple .exe files, the installation prompts the end user to browse to the setup launcher file. If the end user identifies the file, the installation continues. Otherwise, the installation ends.
    c. The setup launcher evaluates the list of features to select which feature prerequisites to install, and it launches their installations as appropriate.
5. The installation finishes making changes on the target system according to the end user's selections.
6. The installation switches from the progress dialog to the Setup Complete Success dialog.

The User Interface for an Installation that Includes InstallShield Prerequisites

If a target system needs one or more setup prerequisites to be installed, the setup launcher typically displays the setup prerequisite dialog before the main installation starts. This setup prerequisite lists all of the nonhidden setup prerequisites that are missing from the target system. When an end user clicks the Install button on this dialog, the setup launcher launches the necessary setup prerequisite installations. If one or more of the setup prerequisites is marked as requiring administrative privileges and the installation is run on a system on which User Account Control (UAC) is enabled, the Install button on this dialog has the shield icon to alert the end user that elevated privileges are required.

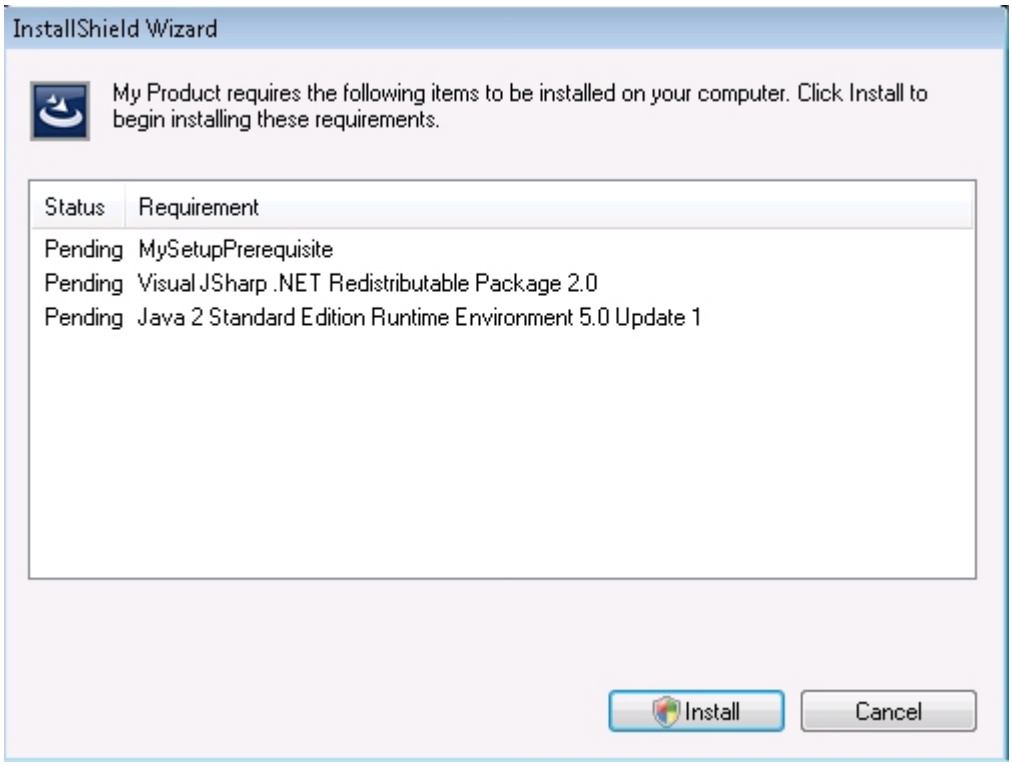


Figure 4-1: Sample Setup Prerequisite Dialog that Shows the List of Setup Prerequisites that Need to Be Installed

If a setup prerequisite is configured to be hidden, it is not listed in the setup prerequisite dialog, but it is still installed. If all of the setup prerequisites in an installation are hidden, the installation displays the setup launcher's standard initialization dialog instead of the setup prerequisite dialog.

If the file that a setup prerequisite installation launches is an .msi package and the prerequisite is marked to show progress, the user interface shows a status bar, along with installation progress messages from Windows Installer, while the prerequisite is being installed.

If a setup prerequisite is configured to be optionally installed by the end user, the setup launcher displays a message box that enables end users to choose whether to install the setup prerequisite.

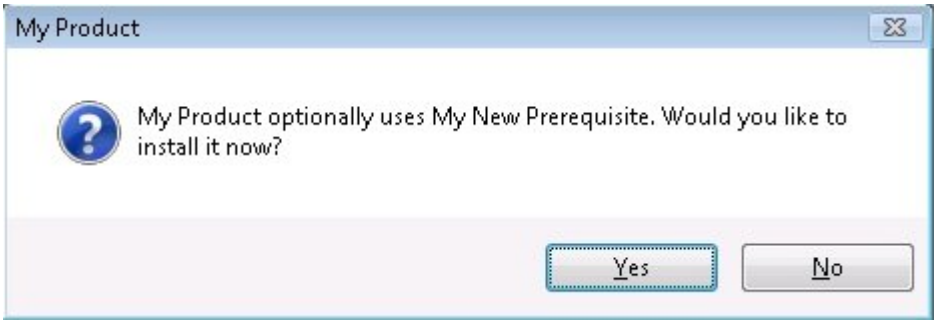


Figure 4-2: Message Box for an Optional Prerequisite

If an installation includes feature prerequisites, the setup launcher does not list them in any prerequisite dialog. However, the user interface does show progress messages if appropriate. In addition, the setup launcher displays the optional prerequisite message box if the feature prerequisite is marked as optional.

Silent Scenarios—Suppressed User Interface

The installation can install setup prerequisites and feature prerequisites even if the installation is run silently. That is, InstallShield prerequisites are supported in any of the following scenarios:

- **Silent setup launcher and visible .msi package**—The user interface for the setup launcher is suppressed, but the user interface for the .msi package is visible. For example, the end user might use the following command-line statement:

  ```
  Setup.exe /s
  ```

  In this scenario, the language selection dialog and the setup prerequisite dialog are not displayed.

- **Visible setup launcher and silent .msi package**—The user interface for the setup launcher is displayed, but the user interface for the .msi package is suppressed. For example, the end user might use the following command-line statement:

  ```
  Setup.exe /v"/qn"
  ```

  In this scenario, the feature selection dialog and all of the other dialogs of the main installation are not displayed. However, the end user can set Windows Installer properties such as `ADDLOCAL`, `ADDSOURCE`, `ADDDEFAULT`, and `ADVERTISE` from the command line to indicate which features should be installed.

- **Silent setup launcher and silent .msi package**—The user interface for the setup launcher and the .msi package are suppressed. For example, the end user might use the following command-line statement:

  ```
  Setup.exe /s /v"/qn"
  ```

  In this scenario, all of the setup launcher and .msi package dialogs are suppressed.

If the UI sequence of the main installation's .msi package is skipped, the setup launcher evaluates Windows Installer properties such as `ADDLOCAL`, `ADDSOURCE`, `ADDDEFAULT`, and `ADVERTISE` to determine if any feature prerequisites should be installed, and it installs feature prerequisites accordingly.

UAC Prompts

Depending on how it is configured, an installation that includes InstallShield prerequisites may prompt for elevated privileges on Windows Vista and later systems at several different points during the installation:

1. When the end user launches the `Setup.exe` file
2. When the `Setup.exe` file launches a setup prerequisite that requires elevated privileges
3. When the `Setup.exe` file launches a feature prerequisite that requires elevated privileges
4. When the Windows Installer begins the Execute sequence of the .msi package

For more information, see Minimizing the Number of User Account Control Prompts During Installation.

Changing the Behavior of InstallShield Prerequisites

***Edition** ▪ InstallShield Premier Edition and InstallShield Professional Edition include the InstallShield Prerequisite Editor, a tool that enables you to change the behavior of an InstallShield prerequisite.*

The InstallShield Prerequisite Editor that is available in the Premier and Professional editions of InstallShield enables you to configure certain prerequisite behavior and configuration settings. For example, with the InstallShield Prerequisite Editor, you can do any of the following tasks:

- Specify whether a prerequisite should be listed in the setup prerequisite dialog at run time.
- Specify whether a prerequisite should be optional.
- Configure an InstallShield prerequisite so that it is installed either before or after any installation of the Windows Installer engine and the .NET Framework.
- Specify command-line parameters that should be passed to the prerequisite when it is launched so that it runs silently.
- Specify whether the prerequisite requires administrative privileges.

Uninstalling an Application Whose Installation Included InstallShield Prerequisites

Your installation may consist of your application plus one or more InstallShield prerequisites. If end users uninstall your application through Add or Remove Programs in the Control Panel, the InstallShield prerequisites are still installed on their machines. If an InstallShield prerequisite installation added an entry to Add or Remove Programs, an end user would be able to remove that InstallShield prerequisite through Add or Remove Programs.

Working with Merge Modules and Objects that Are Included in Installation Projects

This section of the documentation offers guidance for working with merge modules and objects from within installation projects.

***Edition** ▪ InstallShield Premier Edition and InstallShield Professional Edition enable you to create your own merge modules.*

Specifying the Directories that Contain Merge Modules

InstallShield lets you specify the locations on your local machine, or on a network, where you are storing merge modules (.msm files). This flexibility enables you to store merge modules in source code control and to share a common set of merge modules with other team members.

InstallShield offers several ways for specifying the search paths for merge modules:

- If you are editing or building from within InstallShield, use the Merge Module Options tab on the Options dialog box—which is displayed when you click Options on the Tools menu—to specify a comma-delimited list of machine-wide folders and current-user folders.
- If you are building from the command line with ISCmdBld.exe, use the `-o` parameter to specify a comma-delimited list of folders.

  If you use an .ini file to specify ISCmdBld.exe parameters, you can use the MergeModulePath parameter in the [Mode] section of your .ini file to specify a comma-delimited list of folders.
- If you are building through MSBuild or Team Foundation Server (TFS), use the MergeModulePath parameter on the InstallShield task. This parameter is exposed as the ItemGroup InstallShieldMergeModulePath when the default targets file is used. To specify multiple paths, use an ordered array of paths.

Instead of using hard-coded paths, you can use path variables in paths, as in the following example:

```
<ISProductFolder>\MergeModules,<ISProjectFolder>\MyCustomMergeModules
```

The Redistributables view lists the names of the merge modules that correspond with the merge modules that are present in the various search paths that are specified on the Merge Modules tab of the Options dialog box. If the same merge module is in multiple search paths, InstallShield shows only the first instance that it encounters. InstallShield first searches each path that is listed in the per-user setting on the Merge Modules tab. Then, InstallShield checks each path that is listed in the machine-wide setting.

At build time, if your project includes one or more merge modules, InstallShield searches the specified locations and includes the appropriate merge modules in your release as needed. If the same merge module is in multiple search paths, InstallShield includes in the build only the first instance that it encounters. It uses the following order to search for merge modules:

1. InstallShield checks each path that is listed in the per-user setting on the Merge Modules tab.
2. InstallShield checks each path that is listed in the machine-wide setting on the Merge Modules tab.
3. InstallShield checks the paths that are specified through the `-o` command-line parameter, the MergeModulePath .ini file parameter, or the MergeModulePath parameter on the InstallShield task.
4. If no paths are specified in any of the aforementioned locations, InstallShield checks the following default directories, in order:
   a. *InstallShield Program Files Folder*\System
   b. *InstallShield Program Files Folder*\Modules\i386
   c. *InstallShield Program Files Folder*\Objects
   d. *InstallShield Program Files Folder*\Modules\i386\Japanese
   e. *InstallShield Program Files Folder*\Modules\i386\German
   f. *Program Files Folder*\Common Files\Merge Modules

Modifying the Object and Merge Module Configurations

Once you have added an object or merge module to your project, you may need to change its configuration.

Task ***To modify the configuration of any object or merge module after you have included it in your project:***

1. In the View List under **Specify Application Data**, click **Redistributables**.
2. Right-click the object or merge module that you want to modify and select either **Change Object Settings** or **Configure merge module**. The appropriate wizard opens, enabling you to modify the configuration.

Merge Module Exclusions and Dependencies

When you add a merge module to your installation, you are accepting conditions that go along with that module. Two such conditions are the module exclusions and dependencies.

Module Dependencies

When a module is created, dependencies are set for it. This means that a module will not work if you do not include its dependencies as well. InstallShield automatically associates a module's dependencies with your installation project if they are stored in your local redistributables gallery. If those modules cannot be found, you need to obtain a copy of each and associate them with your installation project in the Redistributables view.

Module Exclusions

Some modules do not operate correctly in the presence of certain other modules. If this is the case, the module's author should list them as exclusions and they should appear in the description window of the Redistributables view. If your newly associated module has exclusions, and those excluded modules have already been associated with your installation, InstallShield silently removes any reference to them in your installation.

Caution ▪ *Any excluded modules that are added to the installation after the excluding module has been added will not be removed; in addition, you will not receive any warning that they are incompatible.*

Overriding a Merge Module's Destination

Although you should not alter a third-party merge module, you can override the destination for an InstallShield-created merge module and some third-party merge modules.

Note ▪ *This procedure redirects only the* `TARGETDIR` *directory in the merge module, or directories that derive directly from* `TARGETDIR`*. If the merge module is configured to send files to a predefined folder (for example, SystemFolder), you cannot override the module's destination.*

Task ***To override a merge module's destination:***

1. In the View List under **Specify Application Data**, click **Redistributables**.
2. Select the check box next to the merge module to add it to your installation.
3. Right-click the module and click **Properties**. The Merge Module Properties dialog box opens.
4. In the **Destination** box, type a destination or select one from the list of predefined destinations.
5. Click **OK**.
6. In the **Conditional Installation** pane, select the feature or features that should contain the merge module.

Troubleshooting Merge Module Issues

If you delete a merge module that is currently associated with your installation, the message [Merge Module Not Found] is displayed, thereby making you aware that the module cannot be included in your installation.

Adding Windows Installer Redistributables to Projects

Although the Windows Installer is built into most versions of Windows, a Windows Installer–based installation may depend on certain functionality that is available in only the latest versions of Windows Installer.

InstallShield enables you to include a redistributable for Windows Installer in your project. The method for adding a Windows Installer redistributable to your project depends on the version of Windows Installer that your installation requires.

Note that Windows Installer 5 and Windows Installer 4 are not available as redistributables.

Windows Installer Distribution

By default, InstallShield creates a `Setup.exe` setup launcher along with your installation package. The setup launcher is required if you want your installation to install the Windows Installer engine.

If you want to include the Windows Installer redistributable in your project, do one of the following:

- **For Windows Installer 4.5**—Add one or more Microsoft Windows Installer prerequisites to your project. InstallShield includes several versions that target different versions of Windows. For more information, see Including Microsoft Windows Installer Prerequisites.
- **For Windows Installer 3.1, 3.0, or 2.0**—The Setup.exe tab for a release in the Releases view is where you specify information such as whether you want to use a `Setup.exe` launcher, whether you want to include one of these versions of the Windows Installer redistributable, and which version of Windows Installer you want to include. To learn more, see Setup.exe Tab.

  As an alternative, you can add one or more Microsoft Windows Installer prerequisites to your project. For more information, see Including Microsoft Windows Installer Prerequisites.

Overview of the Installation Process

At run time, `Setup.exe` determines if Windows Installer is already installed on the target system. If the Windows Installer is found on the target system and it meets the minimum version requirement, it launches your installation package. If Windows Installer is not installed, or a more recent version needs to be installed, `Setup.exe` installs Windows Installer and then launches your installation package. Note that the system may need to be restarted for Windows Installer to be updated.

Including Microsoft Windows Installer Prerequisites

InstallShield includes InstallShield prerequisites for several versions of Windows Installer. You can use the Redistributables view to add these InstallShield prerequisites to your project.

Adding .NET Framework Redistributables to Projects

If your product requires that the .NET Framework be installed on the target system, you can add the .NET Framework redistributable to your project. If the target system does not have the .NET Framework, it is installed during your installation.

You can also include redistributables for .NET Framework language packs in your project. The language packs contain translated text, such as error messages, for languages other than English.

The method for adding the .NET Framework and .NET Framework language packs to your project depends on the version of .NET Framework that your application requires.

Note ▪ *Some .NET Framework versions include earlier .NET Framework versions:*

- *.NET Framework 3.5 includes .NET Framework 3.0 SP1 and .NET Framework 2.0 SP1.*
- *.NET 3.0 Framework SP1 includes .NET Framework 2.0 SP1.*
- *.NET 3.0 Framework RTM includes .NET Framework 2.0 RTM .*

If you want to include .NET support in a project, do one of the following:

- **For .NET Framework 4.5 Full, 4.5 Web, 4.0 Full, 4.0 Client, 3.5 SP1, 3.5, 3.0 SP1, 3.0, 2.0 SP2, or 2.0 SP1 redistributables**—Add the appropriate Microsoft .NET Framework prerequisite.

  For more information, see Including the Microsoft .NET Framework and Microsoft .NET Framework Language Pack Prerequisites.

- **For .NET Framework 2.0, 1.1, or 1.0 redistributables**—Configure the .NET settings for the release through the .NET/J# tab in the Releases view.

Obtaining InstallShield Prerequisites

Note that some of the InstallShield prerequisites are not installed with InstallShield. You may need to download them. For more information, see Obtaining Updates for InstallShield.

Including the Microsoft .NET Framework and Microsoft .NET Framework Language Pack Prerequisites

InstallShield includes InstallShield prerequisites for some versions of the .NET Framework and the .NET Framework language packs. You can include these InstallShield prerequisites in your projects if you want to redistribute these versions of the .NET Framework and the language packs.

Following is a list of the .NET Framework redistributables that are available as InstallShield prerequisites. The associated language pack prerequisites are included if available.

- Microsoft .NET Framework 4.5. (One InstallShield prerequisite is for the full package, and one is for the Web package. The Web package is smaller in size, but it requires an Internet connection on the target system at run time.)
- Microsoft .NET Framework 4 Full, which installs the .NET Framework runtime and associated files that are required to run and develop applications that target the .NET Framework 4. (One InstallShield prerequisite is for the full package, and one is for the Web Download package. The Web Download package is smaller in size, but it requires an Internet connection on the target system at run time.)
- Microsoft .NET Framework 4 Client, which installs the .NET Framework runtime and associated files that are required to run most client applications. (One InstallShield prerequisite is for the full package, and one is for the Web Download package. The Web Download package is smaller in size, but it requires an Internet connection on the target system at run time.)
- Microsoft .NET Framework 3.5 (One InstallShield prerequisite is for the full package, and one is for the Web Download package. The Web Download package is smaller in size, but it requires an Internet connection on the target system at run time.)
- Microsoft .NET Framework 3.0 SP1 (This is a Web Download package, which requires an Internet connection on the target system at run time.)
- Microsoft .NET Framework 3.0
- Microsoft .NET Framework 2.0 SP2
- Microsoft .NET Framework 2.0 SP1

Tip ▪ *For information on other versions of the .NET Framework redistributables, see Adding .NET Framework Redistributables to Projects.*

These InstallShield prerequisite installations are run in silent mode. Therefore, the language in which the .NET Framework installations are run is not an issue.

The InstallShield prerequisite installations determine if the corresponding version of the .NET Framework is already installed on the target machine by checking the Install or InstallSuccess value data for a particular HKEY_LOCAL_MACHINE key. For more details, you can click the InstallShield prerequisite in the Redistributables view and note the conditions that are defined in the details pane.

Including the DirectX 9.0 Object

DirectX provides supporting API libraries for multimedia applications and hardware, including the latest graphics cards. If your product requires DirectX to be installed on the target system, you can add the DirectX object to your project. If the target system does not have DirectX, it is installed during your installation.

After installation, the DirectX runtime cannot be uninstalled. DirectX is a system component; end users cannot uninstall it without reinstalling the operating system.

Tip ▪ *The DirectX object is not installed with InstallShield; you need to download it. To learn more, see Downloading Redistributables to Your Computer.*

Redistributable Files

The DirectX objects install all DirectX 9.0c core and optional components.

Including the DirectX Object in Projects

When you add the DirectX object to an Express project, InstallShield launches the DirectX Object Wizard.

You can use the DirectX object in compressed or uncompressed installations; the DirectX Object Wizard lets you specify whether the DirectX files should be in a folder in the Disk1 folder, or they should be streamed into the .msi file:

- If you specify that the files should be in a folder in the Disk1 folder, InstallShield creates a DirectX folder for your installation at build time and places it in the Disk1 folder of your release. InstallShield lists the DirectX folder in the Disk1 area of the Setup Files view.
- If you specify that the files should not be in the Disk1 folder, InstallShield embeds the files in your installation's .msi file.

Note ▪ *The custom action that launches the DirectX installation is sequenced in the Execute sequence and run in deferred system context so that it can be run with elevated privileges on Windows Vista and later systems.*

Updating the DirectX Object Files for Express Projects

If you obtain updates for any of the DirectX files and you want to include them in your DirectX object, place them in the appropriate InstallShield Program Files subfolder with the other DirectX files. The location is:

InstallShield Program Files Folder\Objects\DirectX9c\Redist

You can also remove files from that folder if your product does not require them and you do not need to include them in your installation. For more information about the DirectX redistributable files, including details about any updates that Microsoft may have released since the current version of InstallShield was released, see the latest DirectX SDK or Microsoft's MSDN Web site.

At build time, InstallShield uses whatever redistributable files are in that DirectX folder to build your release.

Identifying Application Dependencies

A file often relies on functions in other files to perform a task. However, you may not be aware of all these other files—known as *dependencies*—when you include your application's files in your installation project. InstallShield offers the following scanning wizards to assist you in identifying and working with these files. You can access these scanners in the Dependencies view.

Table 4-7 ▪ Scanning Wizards Available in InstallShield

Scanner	Function
Static Scanning Wizard	Looks at portable executable (.exe, .ocx, .com, .tlb, .hlp, and .chm) files in your project and checks for any dependencies that they may require.
Dynamic Scanning Wizard	Monitors your system while an executable file is running and checks for .dll or .ocx files that may be required by the executable file.

If you use the Static and Dynamic scanning wizards, you can specify files that you want to be included or excluded automatically any time that you perform a static or dynamic scan through InstallShield. For more information, see Filtering Files in Dependency Scanners.

Static Scanning

The Static Scanning Wizard enables you to scan the files already added to your project for any dependencies that they may require. This wizard scans all portable executable files (.exe, .dll, .ocx, .sys, .com, .drv, .scr, and .cpl files) in your project and checks for dependencies that they require. The wizard displays a list of the dependencies that it finds, and it lets you specify whether you want to include each one in your project.

The new files added to your project are added to the same feature as the file that depends upon them, thereby ensuring that they are installed when needed.

To learn more about this wizard, see Static Scanning Wizard.

Dynamic Scanning

The Dynamic Scanning Wizard is an easy-to-use tool that monitors your system while an executable file runs. The wizard displays a list of .dll and .ocx files that may be required by the executable file, and it lets you specify whether you want to include each one in your project.

The executable file you want to scan can already be included to your project, or it can be added by the wizard.

To learn more about this wizard, see Dynamic Scanning Wizard.

Reviewing Dependency Scanner Results

The Static Scanning Wizard and the Dynamic Scanning Wizard scan your project to help you identify other files that your product may require. Both of these wizards displays a list of possible files and merge modules that you may need to add to your project. Review the scan results carefully, and determine whether it is appropriate to add each specified file and merge module to your project.

Both of the scanners works differently to identify dependencies. In some cases, one of the scanners may identify a dependency that the other does not identify. Therefore, you may want to try using both the static and dynamic scanning wizards for a more complete list of potential dependencies. In some scenarios, a dependency scanner may identify as a dependency a file or merge module that is not required by your product. If that occurs, you can exclude that file or merge module, since the wizards let you include or exclude each identified dependency as needed. In addition, InstallShield enables you to specify on a machine-wide basis any files that you want to be included or excluded automatically any time that you perform a static or dynamic scan through InstallShield. For more information, see Filtering Files in Dependency Scanners.

To obtain the best results when you are trying to identify dependencies, it is recommended that you thoroughly test your product and its installation on a clean machine. If your product does not behave as expected, determine whether any dependencies are missing from the machine, and if so, whether they should be included in the installation.

Filtering Files in Dependency Scanners

When you run the Static and Dynamic scanning wizards, you may find that they list as dependencies certain files that you do not want added to your installation. To avoid having these files added every time you run the scanner, you can edit `Filters.xml`. This file enables you to specify any files that you want the scanners to ignore or include.

`Filters.xml` is in the following location:

`InstallShield Program Files Folder``\Support`

The file must remain in this location after you edit it to ensure that the scanners work properly.

Tip ▪ *You can also use the* `Filters.xml` *file to control which registry items should be excluded during COM extraction. For more information, see Filtering Registry Changes for COM Extraction.*

Excluding Files

The <Exclude> element in the `Filters.xml` file is where you add subelements for each of the files that you want the scanners to exclude. Any files that are listed here will not be added to your installation project by the scanners.

By default, the <Exclude> element has subelements for common system files that are present on all Windows–based machines.

Including Files

The <Include> element in the `Filters.xml` file gives you the ability to override individual files that are subelements of the <Exclude> element. Any files that are listed in subelements of the <Include> element will be added to your installation project by the scanners, even if the files are also listed in subelements of the <Exclude> element.

Note ▪ *The following vital operating system files are never recognized by the scanner, even if you add them as subelements of the <Include> element:*

- *`kernel32.dll`*
- *`ntdll.dll`*

- *user32.dll*
- *gdi32.dll*
- *advapi32.dll*
- *shell32.dll*
- *ole32.dll*

Specifying Files in the <Exclude> and <Include> Elements

If you want to list a file under the <Exclude> or <Include> elements, you must add the file as a subelement. Following is a sample of a properly formatted subelement:

```
<File name="myfile.dll" path="[SystemFolder]" We="needthis"/>
```

Table 4-8 ▪ Recognized Attributes for the <File> Subelement

Attribute	Description
name	This attribute must be lowercase. The value of this attribute (for example, **myfile.dll** in the above example) indicates the name of the file that you want to include or exclude.
path	This attribute is optional. The value of this attribute (for example, **[SystemFolder]** in the above example) indicates the path of the file.

Any other attributes are optional and are not recognized by the scanners. You may want to add additional attributes—such as the **We** attribute in the example above and the corresponding **"needthis"** value—to identify why an item is being excluded or included.

Important ▪ *Ensure that your XML code is well formed; if its not well formed, all of the filters fail. In most cases, you can identify improperly formed XML code by opening the* `Filters.xml` *file in Internet Explorer. You should be able to expand and contract the <Filters>, <Include>, and <Exclude> elements; if you cannot, check the code for errors.*

If you add subelements to the <Exclude> or <Include> elements, be sure that you do not place them within the commented-out section, since InstallShield ignores that area of the `Filters.xml` *file.*

The following sample XML code shows the format of the `Filters.xml` file:

```
<Filters>
 <Include>
  <!--Instructions on how to add files to this element.
  -->
  <File name="mfc42.dll" We="needthis"/>
 </Include>
 <Exclude>
  <!--Instructions on how to add files to this element.
  -->
  <Registry key="HKEY_CLASSES_ROOT\Interface\{00020404-0000-0000-C000-000000000046}"/>
  <File name="12520437.cpx" path="[SystemFolder]" wrp="4.0-10.0" />
  <File name="12520850.cpx" path="[SystemFolder]" wrp="4.0-10.0" />
 </Exclude>
</Filters>
```

Registering COM Servers

Most applications require certain COM servers in order to operate properly. In order for the operating system to recognize these COM servers, you need to register them.

InstallShield supports two methods for registering a COM server on a target machine:

- The COM information can be extracted from the file and used to register the COM server during the installation.
- The file can be self-registered if it supports self-registration. Note that self-registration is not as reliable as having Windows Installer register and unregister the file with extracted COM information.

The first method listed—extracting the COM information—is the recommended method.

Important ▪ *Some applications, like WinRunner, insert hook .dll files into the COM extraction engine. This causes COM extraction to fail and displays the following message: "ISRegSpy detects following module %1 hooked into this process, which causes ISRegSpy to malfunction. You need to shut the application down and restart COM extraction." If you encounter this message, shut down the application and restart COM extraction, as the dialog box instructs.*

Do not select the self-registering property for .exe files that are not self-registering. To self-register an .exe file, you need to launch the .exe file with the `/regserver` *command. However, if the .exe file does not support the command-line switch, the .exe file will be launched during extraction at build time.*

Task ***To register a COM server with InstallShield:***

1. In the View list under **Specify Application Data**, click **Files**.
2. Right-click the file you want to register and then click **Properties**. The **Properties** dialog box opens.
3. Click the **COM & .NET Settings** tab.
4. In the **Registration Type** list, select the type of registration that you want to perform on the file.
5. Click **OK**.

Filtering Registry Changes for COM Extraction

To prevent InstallShield from extracting undesired COM data from a COM server (either at build or design time), you can edit the `Filters.xml` file and specify the registry keys to be excluded. `Filters.xml` is in the following location:

`InstallShield Program Files Folder``\Support`

The file must remain in this location after you edit it to ensure that COM extraction works properly.

Tip ▪ *You can also use the* `Filters.xml` *file to control which files should be included or excluded during dependency scanning. For more information, see Filtering Files in Dependency Scanners.*

Excluding Registry Keys from COM Extraction

The <Exclude> element in the `Filters.xml` file is where you add subelements for each of the registry keys that you want the COM extraction process to exclude. Any keys that are listed here will not be uninstalled when your product is uninstalled.

By default, the <Exclude> element has subelements for common system registry keys that are required.

Specifying Registry Keys in the <Exclude> Element

If you want to list a key under the <Exclude> element, you must add the key as a registry subelement.

Following is a sample of a properly formatted registry subelement that blocks changes to an InprocServer32 registry key, all of its values, and all of its subkeys:

```
<Registry key="HKEY_LOCAL_MACHINE\SOFTWARE\Classes\CLSID\{00000231-0000-0010-8000-00AA006D2EA4}\InprocServer32"/>
```

Following is a sample of a properly formatted registry subelement that blocks changes to only the default value of an InprocServer32 registry key:

```
<Registry key="HKEY_LOCAL_MACHINE\SOFTWARE\Classes\CLSID\{00000231-0000-0010-8000-00AA006D2EA4}\InprocServer32" value=""/>
```

Following is a sample of a properly formatted registry subelement that blocks changes to only the ThreadingModel value name for an InprocServer32 registry key:

```
<Registry key="HKEY_LOCAL_MACHINE\SOFTWARE\Classes\CLSID\{00000231-0000-0010-8000-00AA006D2EA4}\InprocServer32" value="ThreadingModel"/>
```

Table 4-9 ▪ Recognized Attributes for the <Registry> Subelement

Attribute	Description
key	This attribute must be lowercase. The value of this attribute (for example, HKEY_LOCAL_MACHINE\SOFTWARE\Classes\CLSID\{00000231-0000-0010-8000-00AA006D2EA4}\InprocServer32 in the above examples) indicates the name of the registry key that you want to filter.
value	This attribute is optional: ● To block changes to an entire registry key, do not include the value attribute. ● To block changes to the default value of the specified key, set the value of this attribute to a null value. ● To block changes to the name of a value of the specified key, set the value of this attribute to the name of that registry value.

Any other attributes for the <Registry> subelement are optional and are not recognized by the COM extraction process. You may want to add additional attributes to identify why an item is being excluded.

Important ▪ *Ensure that your XML code is well formed; if its not well formed, all of the filters fail. In most cases, you can identify improperly formed XML code by opening the* `Filters.xml` *file in Internet Explorer. You should be able to expand and contract the <Filters>, <Include>, and <Exclude> elements; if you cannot, check the code for errors.*

If you add subelements to the <Exclude> or <Include> elements, be sure that you do not place them within the commented-out section, since InstallShield ignores that area of the `Filters.xml` *file.*

The following sample XML code shows the format of the `Filters.xml` file:

```
<Filters>
 <Include>
  <!--Instructions on how to add files to this element.
  -->
 </Include>
 <Exclude>
  <!--Instructions on how to add files to this element.
  -->
  <Registry key="HKEY_CLASSES_ROOT\Interface\{00020404-0000-0000-C000-000000000046}"/>
  <File name="12520437.cpx" path="[SystemFolder]" wrp="4.0-10.0" />
  <File name="12520850.cpx" path="[SystemFolder]" wrp="4.0-10.0" />
 </Exclude>
</Filters>
```

Configuring the Target System

Every installation changes the target system in some way. The simplest installations might only copy files. More in-depth installations make registry changes, create file associations, edit .ini files, create shortcuts, configure ODBC resources, use environment variables, and install and start Windows services. For more information on how to configure the target system, refer to this section of the documentation.

Creating Shortcuts and Program Folders

Shortcuts and program folders offer quick access to your installed application. You can configure your installation so that it creates shortcuts and program folders on the desktop, the Start menu, and various other locations on the target system. The shortcuts and folders are created on the target system only if the feature to which they belong is selected for installation.

Types of Shortcuts

InstallShield offers several types of shortcuts:

Table 4-1 ▪ Types of Shortcuts

Shortcut Type	Description
New Shortcut	Creates a standard shortcut to a file that exists in your project.
New Advertised Shortcut	Creates an advertised shortcut. The feature's files are not installed to the target system until the end user launches the shortcut. ***Note ▪*** *An advertised shortcut can be created for a file only if it is designated as a portable executable file. You can indicate which files you want to be treated as portable executable files on the File Extensions tab of the Options dialog box.*
New Shortcut to Preexisting File	Creates a shortcut to a file that already exists on the target system. For example, you could use this to create a shortcut to Internet Explorer or to UNC paths.
New Folder	Creates a program folder. You can create a program folder if you want your shortcut to appear under your company name, for example.
New Uninstall Shortcut	Creates a shortcut that automatically starts the uninstallation process for a product.

Creating Shortcuts

Before you create a shortcut, you should create the feature that contains the file to which the shortcut points.

Task ***To create a new shortcut:***

1. In the View List under **Configuring the Target System**, click **Shortcuts/Folders**.
2. In the **Shortcuts** explorer, right-click one of the destination directories and then click the appropriate command. For a list of available commands, see Types of Shortcuts.

   InstallShield adds a new shortcut with the default name **NewShortcutN** (where *N* is a successive number).
3. Enter a new name, or right-click it later and click **Rename** to give it a new name.
4. Configure the shortcut's settings.

For details about each of the settings that you can configure for shortcuts and folders, see:

- Shortcut Settings
- Folder Settings

Note ▪ *You can create a program folder if you want your shortcut to appear under your company name, for example. When you have created a folder for your shortcut, you can create the shortcut by right-clicking your new folder and then clicking New Shortcut.*

You cannot create shortcuts to a dynamically linked file. For more information, see Limitations of Dynamic File Linking.

Specifying the Icon for a Shortcut

InstallShield enables you to specify the icon that should be used for a shortcut that is created on the target system at run time.

Task ***To specify the icon for a shortcut:***

1. In the View List under **Configuring the Target System**, click **Shortcuts/Folders**.
2. In the **Shortcuts** explorer, click the shortcut whose icon you want to specify. The shortcut's settings are displayed in the right pane.
3. In the **Icon File** setting, specify the file that contains the icon for the shortcut that you are creating. You must specify an .ico file or the executable file (.dll or .exe) that contains the icon resource. You can either type the fully qualified path to the file that contains the icon, or click the ellipsis (...) button to browse to it.
4. If the icon file that you specify contains more than one icon resource, enter the index in the **Icon Index** setting.

   A nonnegative integer refers to the order of the icon resources in the executable file. For example, 0 refers to the first icon in the file, 1 refers to the second icon, and 2 refers to the third icon.

InstallShield changes the icon that is displayed for the shortcut in the Shortcuts explorer to the one that you specified, unless the shortcut is for a pre-existing file on the target system.

If the shortcut is for a pre-existing file, the icon file is not known until run time. Therefore, InstallShield shows the following icon for each shortcut in the Shortcuts explorer, instead of displaying the icon that is used at run time on target systems.

InstallShield also uses this icon for a shortcut in the Shortcuts explorer if the file that is selected in the Icon File setting does not contain an icon.

Project ▪ *Since Windows Installer requires a separate icon when the feature is advertised, InstallShield extracts the icon from any executable file that you specify.*

Tip ▪ *You can also right-click the icon in the Shortcuts/Folders view and then click Change Shortcut icon to specify a different shortcut icon. InstallShield updates the value in the Icon File and Icon Index settings with the values that you specify using this method.*

Basing a Shortcut on the Source Media

Task ***To base a feature's shortcut on the source media:***

1. In the View List under **Organize Your Setup**, click **Features**.
2. In the **Features** explorer, click the feature.
3. Set the **Remote Installation** property to **Favor Source**.
4. Build your release as an uncompressed CD-ROM.
5. Create an advertised shortcut to the feature.

With these settings, all files in the feature remain on the CD and are *not* installed locally. When the end user launches the advertised shortcut, the Windows Installer launches (invisible to the end user) and runs the ResolveSource action. This action determines the value of `SourceDir`. Because the Build is uncompressed, `SourceDir` contains the path to the CD drive and it resolves to the file on the CD.

Specifying a Keyboard Shortcut for Accessing a Shortcut

Keyboard shortcuts—also sometimes called hot keys—enable you to perform tasks quickly by pressing a combination of keys, such as CTRL+ALT+A, instead of using the mouse. You can assign a keyboard shortcut to your product's shortcut so that end users can press the appropriate hot keys to launch the shortcut.

Caution ▪ *It is recommended that you avoid configuring keyboard shortcuts for your shortcuts because they may conflict with existing keyboard shortcuts on the target system.*

Task ***To assign a keyboard shortcut to a shortcut in your project:***

1. In the View List under **Configuring the Target System**, click **Shortcuts/Folders**.
2. In the **Shortcuts** explorer, select the shortcut for which you are specifying a hot key.
3. In the **Hot Key** setting, click the ellipsis button (**...**). The **Hot Key** dialog box opens.
4. Press the keyboard shortcut that you want to use for this shortcut.
5. Click **OK**.

In the Hot Key setting, InstallShield displays the appropriate decimal value that represents the combination of keys that you pressed.

For example, if your key combination is CTRL+ALT+A, InstallShield displays 1601 in this setting. This number is obtained by combining the hex value of CTRL (200) and the hex value of ALT (400) with the logical Or operator. Then, that number (600) is added to the hex value for the A key (41) and converted to a decimal value. In this example, the number that is converted to decimal is 641. After the conversion, it is 1601.

Renaming Shortcuts

When you create a shortcut, your shortcut appears with a default internal name. This name is not displayed to end users, but you can change it to something that is relevant to your project.

Task ***To rename a shortcut:***

1. In the View List under **Configuring the Target System**, click **Shortcuts/Folders**.
2. In the **Shortcuts** explorer, right-click the shortcut that you want to rename and click **Rename**.
3. Type the new name.

Creating an Uninstallation Shortcut

Although it is possible to create a shortcut for uninstallation, the recommended way for users to uninstall products is through Add or Remove Programs in the Control Panel. If you do want to create an uninstallation shortcut, follow the procedure described below.

Task ***To create a shortcut that automatically starts your product's uninstallation process:***

1. In the View List under **System Configuration**, click **Shortcuts/Folders**.
2. In the **Shortcuts** explorer, right-click the folder that should contain the uninstallation shortcut and then click **New Uninstall Shortcut**. InstallShield creates a new shortcut with the default name **Uninstall**.
3. Type a name for the shortcut. To rename the feature, right-click it and click **Rename**.
4. Configure the settings as needed:

Configuring the Appearance of a Desktop App's Tile on the Start Screen

Windows 8 introduced a grid of application tiles to the Start screen, replacing the usual list of shortcuts, and also presented tiles in place of shortcuts. InstallShield supports customizing the appearance of a desktop app's tile on the Start screen. The following tile configuration settings are available:

- A toggle between light or dark text when including the app name on medium-sized (150x150) tiles
- Choice of tile background color
- Option to use custom tile images (small: 70x70 and medium:150x150)
- Preference to show or hide the app name on medium-sized tiles

The **Tile Configurations** node appears in the **Shortcuts/Folders** view. Any applicable tile configurations are listed.

Note ▪ *While there is a relation between shortcuts and tiles, they are different. You can configure shortcuts individually, and even create multiple shortcuts to the same .exe file that use different icons; however, you can only create one tile configuration per .exe file.*

Task ***To configure the appearance of a desktop app's tile on the Start screen:***

1. In the View List under **Configure the Target System**, click **Shortcuts/Folders.**
2. Right-click **Tile Configurations** and choose the **Add Tile Configuration** context menu option.

   Note ▪ *This step assumes an .exe file has been added to the project. If an .exe file does not exist in the project, the* ***Add Tile Configuration*** *context menu option is disabled.*

   Tip ▪ *Alternatively, you can right-click the shortcut that targets the .exe file for which you want to configure a Start screen tile and then choose the* ***Configure Tile*** *context menu option. A new tile configuration is added (and selected). If a configuration has already been added for the shortcut, the* ***Configure Tile*** *option is disabled.*
3. Use any of the settings available for **Tile Configurations** to customize the appearance of a desktop app's tile on the Start screen. For additional information regarding available settings, refer to Tile Configuration Settings.

Editing the Registry

The Windows registry is a system-wide database that contains configuration information used by applications and the operating system. The registry stores all kinds of information, including the following:

- Application information such as company name, product name, and version number
- Path information that enables your application to run
- Uninstallation information that enables end users to uninstall the application easily without interfering with other applications on the system

- System-wide file associations for documents created by an application
- License information
- Default settings for application options such as window positions

Keys, Value Names, and Values

The registry consists of a set of keys that are arranged hierarchically under the Computer explorer (or the My Computer explorer, depending on the operating system of the target system). Just under Computer are several root keys. An installation can add keys and values to any root key of the registry. The root keys that are typically affected by installations are:

- HKEY_LOCAL_MACHINE
- HKEY_USERS
- HKEY_CURRENT_USER
- HKEY_CLASSES_ROOT

A key is a named location in the registry. A key can contain a subkey, a value name and value pair, and a default (unnamed) value. A value name and value pair is a two-part data structure under a key. The value name identifies a value for storage under a key, and the value is the actual data associated with a value name. When a value name is unspecified for a value, that value is the default value for that key. Each key can have only one default (unnamed) value.

Note that the terms key and subkey are relative. In the registry, a key that is below another key can be referred to as a subkey or as a key, depending on how you want to refer to it relative to another key in the registry hierarchy.

InstallShield Projects and the Registry

InstallShield includes the Registry view to help you with the task of modifying the end user's registry. Use this view to create keys and values in much the same way that you use the Windows Registry Editor.

All registry data must be associated with a feature. If the feature is selected for installation, the registry data associated with that feature is set up on the target system.

Caution ▪ *It is important not to modify or delete registry keys indiscriminately because the registry is a vital part of the Windows operating system, and the system may fail to function if vital registry keys are altered.*

HKEY_LOCAL_MACHINE\Software vs. HKEY_LOCAL_MACHINE\Software\Wow6432Node

If you add registry data to the HKEY_LOCAL_MACHINE\SOFTWARE (32-Bit) node in the Registry view, Windows Installer installs that data under the HKEY_LOCAL_MACHINE\Software key on 32-bit target systems but under the HKEY_LOCAL_MACHINE\Software\Wow6432Node key on 64-bit target systems.

If you add registry data to the HKEY_LOCAL_MACHINE\SOFTWARE (64-Bit) node in the Registry view, InstallShield creates a 64-bit Windows Installer package for your project at build time; this 64-bit Windows Installer package cannot be run on 32-bit target systems. In this scenario, Windows Installer installs the registry data under the HKEY_LOCAL_MACHINE\Software key on 64-bit target systems.

Thus, if you are creating a 64-bit installation and you want to install registry entries to 64-bit registry locations (under HKEY_LOCAL_MACHINE\Software instead of HKEY_LOCAL_MACHINE\Software\Wow6432Node), add the entry to the SOFTWARE (64-Bit) node, or a subnode. If you are creating a 32-bit installation, avoid putting any data under the SOFTWARE (64-Bit) node.

To learn more, see Challenges of Supporting Both 32-Bit and 64-Bit Target Operating Systems.

Tip ▪ *To see how a 32-bit application views the registry on a 64-bit system, launch the 32-bit version of the Registry Editor (the* `regedit.exe` *file in the SysWOW64 folder).*

Filtering Registry Entries by Feature

The Registry view includes a Feature list. The Feature list contains your project's hierarchy of features and subfeatures, and it enables you to select a feature whose registry data you want to display in the view.

If the feature hierarchy contains a subfeature, selecting a parent feature displays only the registry entries in that feature. It does not display the registry entries in the subfeature.

Viewing All Registry Entries in Your Project

To view all of the registry entries in your installation, select the All Application Data option in the Feature list.

You can modify, rename, or delete registry keys and values while filtering the view by All Application Data.

When you click a registry key in the Destination computer's Registry view pane of the Registry view, InstallShield displays all of the registry data for that key in the lower-right pane of the Registry view.

If you have not set a value for a key, no registry data is displayed for that key when you select All Application Data.

Creating a Registry Key

Task ***To specify a registry key to be created on the target system when a feature is installed:***

1. In the View List under **Configure the Target System**, click **Registry**.
2. In the **Feature** list, select the feature with which you want to associate the new key.
3. In the **Destination computer's Registry view** pane, click the registry key under which you want to create your new key.
4. In the **Destination computer's Registry view** pane, right-click a registry key, point to **New**, and then click **Key**.

InstallShield adds a new key with the name **New Key-#***n* (where *n* is a successive number). Enter a meaningful name to rename the key, or right-click the key and select Rename to give it a new name later. Your new key is created with an empty default string value.

To modify the value name and data, see Creating Registry Values.

Tip ▪ *If you are creating a 64-bit installation and you want to install registry entries to 64-bit registry locations (under HKEY_LOCAL_MACHINE\Software instead of HKEY_LOCAL_MACHINE\Software\Wow6432Node), add the entry to the SOFTWARE (64-Bit) node, or a subnode. Note that a 64-bit installation cannot be run on 32-bit target systems. To learn more, see Challenges of Supporting Both 32-Bit and 64-Bit Target Operating Systems.*

Dragging and Dropping Registry Entries to Create Registry Keys

The quickest way to add registry entries to your installation project is to drag them from one of the source panes in the Registry view and drop them into one of the destination panes. When you drop an entire key onto the **Destination computer's Registry view** pane, all of that key's subkeys and values are added to the selected feature.

Task ***To drag and drop a registry entry from the source computer to the destination computer:***

1. In the View List under **Configure the Target System**, click **Registry**.
2. In the **Feature** list, select the feature with which you want to associate the new key.
3. In the **Destination computer's Registry view** pane, click the registry key under which you want to create your new key.
4. In the **Source computer's Registry view** pane, locate the key that you want to include in your installation and drag it to a destination folder in the **Destination computer's Registry view** pane. If you drop an entire key onto the destination pane, all of that key's subkeys and values are added to your installation.

Tip ▪ *If you are creating a 64-bit installation and you want to install registry entries to 64-bit registry locations (under HKEY_LOCAL_MACHINE\Software instead of HKEY_LOCAL_MACHINE\Software\Wow6432Node), add the entry to the SOFTWARE (64-Bit) node, or a subnode. Note that a 64-bit installation cannot be run on 32-bit target systems. To learn more, see Challenges of Supporting Both 32-Bit and 64-Bit Target Operating Systems.*

Using the Context Menu to Drag and Drop Keys

You can use the context menu to move multiple keys and values at one time. Right-click a registry entry, drag it to a destination, and click a command on the context menu.

Table 4-2 ▪ Commands Available from the Registry Entry Context Menu

Option	Description
All keys & values	Adds all selected keys, subkeys, and values.
Key and its values only	Adds only the selected key and the key's values. No subkeys are added.
Only this key	Adds only the selected key, not any of its subkeys or values.
Cancel	Ends the drag and drop operation without making any changes.

Viewing Both the 32- and 64-Bit Areas of the Source Machine's Registry on 64-Bit Development Systems

If you are using InstallShield on a 64-bit development system, the Registry view in InstallShield displays both the 32-bit and 64-bit areas of your machine's registry:

- HKEY_LOCAL_MACHINE\Software
- HKEY_LOCAL_MACHINE\Software\Wow6432Node

This support enables you to drag and drop entries from those source areas to the appropriate areas in the destination pane of this view when you are configuring registry data changes for your project.

Note that if you want your installation to install registry data to a 64-bit area of the registry on 64-bit target systems without having it redirected to a 32-bit area, you must place the registry data in the HKEY_LOCAL_MACHINE\SOFTWARE (64-Bit) node in the destination pane in the Registry view. Simply dragging 64-bit data from the source panes in the Registry view to a non-64-bit location in one of the destination panes of the view does not mark the component as 64 bit. For more information, see Challenges of Supporting Both 32-Bit and 64-Bit Target Operating Systems.

Importing Data From Another Machine

A limitation of the drag-and-drop procedure is that it works only if the registry entries exist on your installation development system. If you have registry data from another machine, you can import that data with the Import REG Files Wizard.

Importing Registry Data from a .reg File

InstallShield enables you to import any existing registry (.reg) files that you may have from other installation projects or that you have created outside InstallShield.

InstallShield can import .reg files created by exporting in Regedit, or files that follow that exact format. Note that InstallShield does not support multiline registry values.

Task ***To import registry data from a .reg file:***

1. In the View List under **Configure the Target System**, click **Registry**.
2. In the **Feature** list, select the feature into which you want to import the .reg file.
3. In the **Destination computer's Registry view** pane, right-click the registry key under which you want your registry data added and then click **Import REG File**. The Registry Import Wizard opens.
4. Follow the directions in the Registry Import Wizard to add the registry data.

When you add registry data to a feature, it is installed on the target system if the associated feature is installed.

Removing Registry Keys

Task ***To remove a registry key:***

1. In the View List under **Configure the Target System**, click **Registry**.
2. In the **Feature** list, select the feature that contains the registry key, or select **All Application Data** to view all of the registry keys for your product.
3. In the **Destination computer's Registry view** pane, right-click the registry key that you want to remove and then click **Delete**.

Creating Registry Values

Task ***To create a new registry value:***

1. In the View List under **Configure the Target System**, click **Registry**.
2. In the **Feature** list, select the feature that contains the registry key to which you want to add a value.
3. In the **Destination computer's Registry view** pane, right-click the key to which you want to add a value, point to **New**, and then click the type of data that you want to register. Available types of registry values are as follows:

   Table 4-3 ▪ Types of Registry Values

Option	Description
Default Value	The key's default value.
String Value	A fixed-length text string.
Binary Value	The value is interpreted and stored as a hexadecimal value.
DWORD Value	Data represented by a number that is four bytes (32 bits) long.
Multi-String Value	Multiple text strings formatted as an array of null-terminated strings, and terminated by two null characters. Selecting this command launches the Multi-Line String Value dialog box.
Expandable String Value	The value is interpreted and stored as an expandable string. According to the Microsoft Developer Network (MSDN), an expandable string registry value is a null-terminated string that contains unexpanded references to environment variables (for example, *%PATH%*).

InstallShield adds a new value with the name **New Value #***n* (where *n* is a successive number). Enter a meaningful name now to rename the value, or right-click the value name and then click Rename to give it a new name later.

Tip ▪ *If you are creating a 64-bit installation and you want to install registry entries to 64-bit registry locations (under HKEY_LOCAL_MACHINE\Software instead of HKEY_LOCAL_MACHINE\Software\Wow6432Node), add the entry to the SOFTWARE (64-Bit) node, or a subnode. Note that a 64-bit installation cannot be run on 32-bit target systems. To learn more, see Challenges of Supporting Both 32-Bit and 64-Bit Target Operating Systems.*

Modifying Registry Value Data

Task ***To modify the data for a registry value:***

1. In the View List under **Configure the Target System**, click **Registry**.
2. In the **Feature** list, select the feature that contains the registry data that you want to modify.
3. In the **Destination computer's registry data** pane, double-click the value that you want to modify. The **Edit Data** dialog box or the **Multi-Line String Value** dialog box opens.
4. Complete the information in the dialog box, and then click **OK**.

Windows Installer properties can be used in your registry value to store information for later use by your product. For example, if you want to store the destination location of your software, enter `[INSTALLDIR]` for your registry value. For more information, see Windows Installer Property Reference.

Note ▪ *To add a value that contains square brackets ([]), you must precede each bracket with a backslash (\) and surround it with an opening and closing bracket. Otherwise, Windows Installer treats the value as a property. For example, if you wanted to write* `[stuff]` *to the registry, use* `[\[]stuff[\]]` *as the value name.*

Removing Registry Values

Task ***To remove a registry value from your project:***

1. In the View List under **Configure the Target System**, click **Registry**.
2. In the **Feature** list, select the feature that contains the registry value that you want to delete.
3. In the **Destination computer's Registry view** pane, click the registry key that has the value you want to delete. All registry values are listed in the **Destination computer's registry data** pane.
4. In the **Destination computer's Registry data** pane, right-click the registry value that you want to remove and then click **Delete**.

Entering Multiple Registry String Values into a Single String

Task ***To enter multiple string values into a single string:***

1. In the View List under **Configure the Target System**, click **Registry**.
2. In the **Feature** list, select the feature that should contain the registry value that you want to add.
3. In the **Destination computer's Registry view** pane, right-click the registry key that should contain the value that you want to add, point to **New**, and then click **Multi-String Value**. The **Multi-Line String Value** dialog box opens.
4. Enter the value information and then click **OK**.

The Type field for this entry appears as `REG_MULTI_SZ`, and the Data field is `File Name Path`.

Referencing an Environment Variable in a Registry Entry

With REG_EXPAND_SZ string values, you can use environment variables for paths that are stored in the registry. These entries require special formatting in order to be recognized by the operating system as environment variables. The format for a REG_EXPAND_SZ value as it appears in the registry is `%TEMP%`. `TEMP` is the standard environment variable for the TEMP directory.

Task ***To reference an environment variable in a registry entry:***

1. In the View List under **Configure the Target System**, click **Registry**.
2. In the **Feature** list, select the feature that should contain the registry value that you want to add.
3. In the **Destination computer's Registry view** pane, right-click the registry key that should contain the value that you want to add, point to **New**, and then click **String Value**.
4. In the **Destination computer's registry data** pane, right-click the value and then click **Modify**.
5. To enter the value data, begin with a pound sign, and then type the environment variable. The environment variable name should be surrounded by percent signs. For example: **`#%TEMP%`**.

The Type field for this entry appears as `REG_EXPAND_SZ`, and the Data field is `%TEMP%`.

Tip ▪ *If you are creating a 64-bit installation and you want to install registry entries to 64-bit registry locations (under HKEY_LOCAL_MACHINE\Software instead of HKEY_LOCAL_MACHINE\Software\Wow6432Node), add the entry to the SOFTWARE (64-Bit) node, or a subnode. Note that a 64-bit installation cannot be run on 32-bit target systems. To learn more, see Challenges of Supporting Both 32-Bit and 64-Bit Target Operating Systems.*

Configuring Permissions for Registry Keys

InstallShield lets you configure settings for securing registry keys for end users who run your product in a locked-down environment. You can assign permissions for a registry key to specific groups and users. For example, you may assign Read, Write, and Delete permissions for a particular registry key to the Administrators group, but only Read permissions for all of the users in a different group.

Task ***To configure the permissions for a registry key:***

1. In the View List under **Configure the Target System**, click **Registry**.
2. In the **Destination computer's Registry view** pane, right-click the registry key and then click **Permissions** button. The **Permissions** dialog box opens.
3. Add, modify, and remove permissions entries as needed. For more information, see Permissions Dialog Boxes for Registry Keys.

Depending on what is selected for the Locked-Down Permissions setting in the General Information view of your project, InstallShield adds permissions data to either the `ISLockPermissions` table or the `LockPermissions` table. To learn more, see Securing Files, Folders, and Registry Keys in a Locked-Down Environment.

Specifying a Primary Key for the Registry Table

Windows Installer requires a unique primary key for each registry key and value you add to the Registry table. To allow you to create registry entries in a completely visual environment, InstallShield assigns a unique name to every entry in the database's Registry table at build time.

You may need to know the entry's primary key when authoring a custom action. InstallShield supports specifying a primary key on a registry key or value in the Registry view.

Task ***To specify a primary key for a registry key or value:***

1. In the View List under **Configure the Target System**, click **Registry**.
2. In the **Feature** list, select the feature that should contain the registry value that you want to add.
3. In the **Destination computer's Registry view** pane, right-click the registry key associated with the primary key that you want to specify, and then click **MSI Value**. The **MSI Value** dialog box opens.
4. Type the name of the key. Since the primary key must be a Windows Installer identifier, the name should contain only letters, numbers, underscores (_), and periods (.), and it must begin with a letter or underscore.

If you do not specify a value, InstallShield generates a unique primary key for this entry in the Registry table.

Registry Flags

Registry flags enable you to control the installation and uninstallation of your registry entries. By default, your registry entries are installed if the feature to which they belong is installed. They are then removed from the target system when that feature is removed. If you would like your registry entries to remain on the target system even after the product has been uninstalled, or if you want to create registry entries only if they do not already exist, you need to set the installation flag for that key.

In InstallShield, installation behavior is set at the subkey level. All values beneath the key must have the same installation and uninstallation behavior.

To change the registry flag of a key, right-click one of your project's keys in the Registry view, and then click any of the commands that are listed in the following table.

Table 4-4 ▪ Registry Flags

Icon	Command	Description
	Automatic	This is the default option for all registry keys. If the key is not already present, the installation creates it. During an uninstallation, if the key is empty, it is removed.
	Install only (+)	 ***Project*** *▪ This registry flag type is available in Express projects.* If the key does not already exist, it is created. The key is left on the target system when the feature to which this key belongs is uninstalled. This option is available only for keys that do not contain subkeys or values.
	Uninstall entire key (-)	 ***Project*** *▪ This registry flag type is available in Express projects.* If this flag applies to an empty key, the key is not created at installation. If this flag applies to a key that contains values and the key does not exist on the target system, the key and values are created at installation. In both cases, the key, all subkeys, and values are removed at uninstallation—even if the subkeys and values were added after the installation.
	Install if absent, Uninstall if present (*)	 ***Project*** *▪ This registry flag type is available in Express projects.* This option is similar to the default behavior (the Automatic registry flag), with one exception. For the Automatic registry flag, if the key is not empty during uninstallation, it is not removed. For the **Install if absent, Uninstall if present (*)** flag, if the registry key is present during uninstallation, the key is removed, regardless of whether its subkeys or values remain.

Setting Install/Uninstall Behavior for Registry Keys

In InstallShield, installation and uninstallation behavior for registry entries is set at the subkey level.

Task ***To set the install/uninstall behavior for a registry key:***

1. In the View List under **Configure the Target System**, click **Registry**.
2. In the **Feature** list, select the feature that corresponds to the registry key whose behavior you are setting.
3. In the **Destination computer's Registry view** pane, right-click the registry key and then click the appropriate behavior.

Install/Uninstall Behavior Options

In InstallShield, installation and uninstallation behavior is set at the subkey level. All values beneath the key must have the same installation and uninstallation behavior. For a list of available options, see Registry Flags.

Handling Registry Entries for a Per-User Installation

Since the current user may not have sufficient privileges for modifying keys under HKEY_LOCAL_MACHINE, you may need to write the entries under HKEY_CURRENT_USER.

When you select HKEY_USER_SELECTABLE in the Registry view, the entries are created under the appropriate registry hive, according to the type of installation and the user's access rights:

- In a per-user installation, meaning that the installation is being run by someone with user-level access privileges, these entries would be made under HKEY_CURRENT_USER.
- In a per-machine installation, meaning that ALLUSERS is not null and that the user is an administrator, the entries would be written under HKEY_LOCAL_MACHINE.

Refreshing the Registry View

Task ***To refresh the registry view:***

Press F12.

Associating a File Extension with an Application File

File associations are registry settings that tell Windows what product to use to open files of a certain type. For example, Windows typically launches `Notepad.exe` when a text file (.txt) is opened.

To view and modify registered file types on your system, open Windows Explorer, and on the Tools menu, click Folder Options. Use the File Types tab of the Folder Options dialog box to see how the file associations are configured.

Similarly, you can identify the application that is associated with a given file by right-clicking the file in Windows Explorer and then clicking Properties.

File associations are stored in both HKEY_LOCAL_MACHINE\SOFTWARE\Classes and HKEY_CURRENT_USER\SOFTWARE\Classes; you can see a merged view of the data under HKEY_CLASSES_ROOT.

Creating File Associations for Your Installation Project

Best-practice guidelines recommend that you create a file association for every nonhidden type of file that is created or used by your product. With the File Extensions view in InstallShield, you can quickly and easily create file associations for your installation project. When an end user installs a feature that contains a file association, the file association is registered on the target machine; an entry is made in the appropriate part of the registry, and the entry links your file type to your application through the ProgID. The ProgID, which is sometimes called a file type's application identifier or tag name, uniquely identifies your application and ensures that your association is recognized by the operating system.

Creating a File Extension Association

You can create a file association that links a file extension to an executable (.exe) file in your installation project. Note that the .exe file must be included in your project before you can create the file association.

Note that you should not use a dynamically linked file to create a file association. With dynamically linked files, the selection defaults to the top file in the dynamically linked folder's file list, so you cannot select a specific file.

Task ***To create a file extension association:***

1. In the View List under **Configure the Target System**, click **File Extensions**.
2. Right-click the **File Extensions** explorer and click **New Extension**. InstallShield creates a new file extension with the default name **ext***n* (where *n* is a successive number). Type your own extension without the dot (for example, enter `txt` instead of `.txt`) to replace the default name.
3. Configure the settings for the extension.

Changing .ini File Data

An initialization (.ini) file is a special text file that contains information that applications and the Windows system use to specify application options at startup or run time. Some .ini files, such as Boot.ini and Wininit.ini, are used by the operating system. You can store profile, configuration, language, font, and device driver information in an .ini file.

Note that it is recommended as a general rule that application settings be stored in the registry, instead of in .ini files.

Format of an .ini File

Information in .ini files is stored under section names enclosed within square brackets, and each item has the form keyname=value, as shown below:

```
[section name]
keyname1=value1
keyname2=value2
keyname3=value3
```

Key data can be a string value or an integer value.

InstallShield Projects and .ini Files

The INI File Changes view in InstallShield enables you to specify changes that should be made to .ini files on the target system during the installation of your product. Although you can edit any .ini file found on the target system, editing system .ini files is not recommended.

Editing an .ini file involves three steps:

1. Create an .ini file reference or import an .ini file.
2. Add a section to the .ini file.
3. Add a keyword to the .ini file.

Adding an .ini File

When you add an .ini file to your project, you are creating a reference to that .ini file in your installation. If the feature associated with the .ini file is installed on the target system, the installation modifies the .ini file according to the settings that you configure in the INI File Changes view.

Task ***To add an .ini file:***

1. Open the **INI File Changes** view.
2. Right-click the **INI Files** explorer and click **Add INI File**.
3. Rename the new .ini file, giving it the name of the file name and extension that you want to edit on the target machine; for example, **`Boot.ini`**.
4. Edit the properties of the .ini file.

After you have created the reference to the .ini file, you can move on to the next step, which is to add a section to your .ini file.

Importing an Existing .ini File

Task ***To import an .ini file that exists on your computer:***

1. Open the **INI File Changes** view.
2. Right-click the **INI Files** explorer and click **Import INI File**. The **Open** dialog box opens.
3. Browse to the .ini file that you want to add, select it, and then click **Open**.

InstallShield adds the .ini file to the INI Files explorer; each section and keyword-value pair in the .ini file is represented by a separate item in the INI Files explorer.

Specifying a Section in an .ini File

Once you have specified the .ini file you would like to edit, you can specify which section of that file you want to change. Each .ini file is divided into one or more sections, and each section contain keywords. Sections are identified by the square brackets surrounding them—[SectionName], for example.

Task ***To specify an .ini file section:***

1. Open the **INI File Changes** view.
2. Add or import an .ini file if you have not already done so.
3. In the **INI Files** explorer, right-click the .ini file for which you want to create a section and then click **Add Section**. InstallShield adds a section item with a folder icon to the **INI Files** explorer.
4. Rename this new section, giving it the name of the section you want to edit in the target .ini file. The square brackets are not needed.

After you have added a section to your .ini file you can add a keyword.

Specifying a Keyword and its Value in an .ini File

The keywords in an .ini file are the lowest level of organization in the .ini file. Keywords store data that must persist between uses of an application.

Once you have added a .ini file to your project and set up one or more sections, you can add keywords to the sections and then configure the keyword's properties. The properties of a keyword include the value for the keyword, as well as the action that should be performed (such as replace a data value or append to an existing data value).

Task ***To add a keyword to an .ini file:***

1. Open the **INI File Changes** view.
2. In the **INI Files** explorer, right-click the section that should contain the keyword, and then click **Add Keyword**. InstallShield adds a keyword item to the **INI Files** explorer.
3. Rename this new keyword, giving it the name of the keyword you want to modify. If it is a new entry, enter the name exactly as you want it to appear in the .ini file.
4. Edit the properties of the keyword.

Configuring ODBC Resources

One of the more complex areas of system configuration involves setting up ODBC drivers, data source names (DSNs), and translators. The ODBC resource must be properly registered on the system with all of the required attributes and, in the case of drivers and translators, install the necessary files, including any installation .dll files. This process is simplified in the ODBC Resources view, in which you can select the drivers, data sources, and translators installed on your development system. You can also add to your project any drivers and data sources that are not available on your development system.

Including an ODBC Resource

Task ***To include an ODBC resource in your installation:***

1. Open the **ODBC Resources** view.
2. In the **ODBC Resources** pane, expand the **Drivers & DSNs** item and/or the **Translators** item.
3. Select the check box for the ODBC resource that you want to include.

Including Additional ODBC Resources

You may need to install ODBC drivers or DSNs that are not already listed in the ODBC Resources view. The following procedures explain how to do so.

Note that only translators that are installed on the development system are listed in the ODBC Resources view. To add to your project a translator that is not listed in this view, you must first install the translator on the development system. Then you can add it to your project through the ODBC Resources view.

Task ***To include a driver that is not listed in the ODBC Resources view:***

1. Open the **ODBC Resources** view.
2. In the **ODBC Resources** pane, right-click **Drivers & DSNs** and then click **Insert Driver**. InstallShield adds a new driver item with the name **NewDriver***n* (where *n* is a successive number). The check box for this new driver is selected by default.
3. Type a new name for the driver.
4. In the **ODBC Attributes & Properties** pane, configure the properties for the driver.

Task ***To include a DSN that is not listed in the ODBC Resources view:***

1. Open the **ODBC Resources** view.
2. In the **ODBC Resources** pane, right-click the driver that should contain the new DSN and then click **New DSN**. InstallShield adds a new DSN item with the name **NewDataResource***n* (where *n* is a successive number). The check box for this new DSN is selected by default.
3. Type a new name for the DSN.
4. In the **ODBC Attributes & Properties** pane, configure the properties for the DSN.

Associating an ODBC Resource with a Feature

Like most of the data in your project, ODBC resources must be associated with a feature. When the feature is installed on the target system, the ODBC resource is installed as a part of the feature. If a resource is associated with more than one feature that is being installed, the resource will be installed only once.

Task ***To associate an ODBC Resource with a feature:***

1. Open the **ODBC Resources** view.
2. In the **ODBC Resources** pane, click a driver, DSN, or translator whose check box is selected.
3. In the **Associated Feature(s)** pane, select one or more check boxes for each feature that should contain the ODBC resource.

Note ▪ *By default, each ODBC resource is associated with the feature Always Install. Since the ODBC Resources explorer requires that at least one feature be selected, you cannot clear the Always Install check box unless you first select the check box for another feature. If only one feature is associated with a ODBC resource, its check box cannot be cleared until you select at least one additional feature.*

Setting the Attributes for an ODBC Resource

You can configure the attributes of any ODBC driver or DSN that you have added to your installation project. See the vendor's documentation for specific information regarding properties and permitted attributes and values.

Task ***To set the attributes for an ODBC resource:***

1. Open the **ODBC Resources** view.
2. In the **ODBC Resources** pane, click a driver, DSN, or translator whose attributes you want to configure. The check box for that resource must already be selected.
3. In the **ODBC Attributes & Properties** pane, configure the resource settings.

Adding a New Attribute to an ODBC Resource

InstallShield enables you to add attributes to any ODBC driver or DSN that you have added to your installation project. See the vendor's documentation for specific information regarding properties and permitted attributes and values.

Adding attributes to a translator is not supported.

Task ***To add a new attribute to a driver or DSN:***

1. Open the **ODBC Resources** view.
2. In the **ODBC Resources** pane, click a driver or DSN that should include the new attribute. The check box for that resource must already be selected.
3. In the **ODBC Attributes & Properties** pane, click the last row of the grid to add a new property and corresponding value.

Removing an Attribute from an ODBC Resource

Task ***To remove an attribute from an ODBC resource:***

1. Open the **ODBC Resources** view.
2. In the **ODBC Resources** pane, click the ODBC resource that contains the attribute that you want to delete. The check box for that resource must already be selected.
3. In the **ODBC Attributes & Properties** pane, right-click the property that you want to remove, and then click **Delete**.

InstallShield removes the corresponding row from the ODBC Attributes & Properties pane.

Using Environment Variables

Environment variables are name and value pairs that can be set on the target system with your installation and can be accessed by your application and by other running programs. Environment variables are stored in the registry.

In the Environment Variables view, you can create new environment variables, modify the values of existing variables, and remove variables. The environment variable creation, modification, or removal takes place when your application is installed—depending on the properties that you set for the environment variable when you added it to your project.

Setting Environment Variables

Task ***To create a new environment variable or modify the value of an existing environment variable:***

1. Open the **Environment Variables** view.
2. Right-click the **Environment Variables** explorer and click **Add Environment Variable**. InstallShield adds a variable with the name **NewEnvironment***n* (where *n* is a successive number).
3. Type the name of the variable that you want to modify, remove, or create.
4. Edit the properties of the environment variable.

Installing and Configuring Windows Services

Edition ▪ *The Express edition of InstallShield includes support for installing a service during installation, and removing the service during uninstallation. It also has support for optionally starting the service after installing it, starting it automatically every time that the system starts, or starting it on demand (when the service is requested through the Service Control Manager).*

The Premier and Professional editions of InstallShield include additional flexibility for services. These editions enable you to start, stop, or delete the service during installation or uninstallation. These editions also let you

configure extended service customization options that are available with Windows Installer 5. In addition, the Premier and Professional editions let you configure a service that is already present on the target system.

Windows services are executable files that Windows–based systems run in the background to manage various system tasks, even if no user is currently logged in. A service is an executable file, but it must be designed as a service; you cannot automatically use an arbitrary executable file as a service. Windows services can be installed to run every time that the system starts or on demand when needed. InstallShield enables you to install new Windows services and configure existing services. Windows has a Services administrative tool with which you can view and configure the services that are installed on a system.

You can use the Services view to specify information about a service that you want to install during installation and remove during uninstallation.

Task ***To install a service:***

1. Add the service executable file to your project. For information on adding files, see Adding Files and Folders to a Project.

   Note that the service must be a single executable file (.exe), since the Windows Installer does not support driver services.

   Also note that the Remote Installation setting for the service's feature must be set to Favor Local. For more information, see Setting a Feature's Remote Installation Setting.
2. In the View List under **Configure the Target System**, click **Services**.
3. Right-click the **Services** node and then click **Add Service**. The **Browse for a Destination File** dialog box opens.
4. Browse to the executable file that you added to your project in step 1. InstallShield adds a new service.
5. Type a new name for the service now, or click it and then press F2 later to rename it.

   The name that you enter must match the name that is shown on the service's Properties dialog box. (To access an installed service's properties: In the Services administrative tool, right-click the service and then click Properties.)
6. Select the service that you added, and then configure the settings that are displayed in the right pane as needed. For information about each of the settings, see Services View.

Note ▪ *You must be familiar with the technical details of your service before you can configure its settings.*

Per-User vs. Per-Machine Installations

Two Windows Installer properties, along with the current user's privileges, affect where the configuration information such as your product's shortcuts and registry entries are stored on a target machine—to the All Users profile or the current user's profile:

- ALLUSERS determines where the configuration information is stored.
- MSIINSTALLPERUSER indicates that the Windows Installer should install the package for only the current user.

The MSIINSTALLPERUSER property is available with Windows Installer 5 and on Windows 7 or Windows Server 2008 R2. Earlier versions of Windows Installer and Windows ignore this property.

ALLUSERS, MSIINSTALLPERUSER, and Windows 7 or Windows Server 2008 R2

If the **ALLUSERS** property is set to 2 and **MSIINSTALLPERUSER** is set to 1, the Windows Installer performs a per-user installation.

During a per-machine installation, the Windows Installer requires elevated privileges, and it directs files and registry entries to per-machine locations. If User Account Control (UAC) is available on the target system, a per-machine installation typically prompts for consent or credentials, depending on the access level of the user. During a per-user installation, the Windows Installer does not prompt for credentials, and it redirects files and registry entries to per-user locations.

For more information, see Single Package Authoring on the MSDN Web site.

Effects of ALLUSERS on Windows Vista and Later

Custom actions that have an in-script execution setting of deferred in system context are used to perform an action with the rights granted to the LocalSystem account on Windows, since the Windows Installer service runs in the system context. Actions not marked as deferred in system context run with user impersonation and have the rights that the user who launches the installation has.

When a per-user installation (that is, one where ALLUSERS is not set) is run, deferred-in-system-context actions run in the same context in which normal deferred or immediate custom actions run, which is with user impersonation. This can potentially cause a run-time issue with the custom action in the following circumstances:

- The user who launches the Windows Installer installation is not an administrator; or the user is running the installation on Windows Vista or later, the user is part of the Administrators group, and the user does not have administrator privileges by default.
- The custom action attempts to modify a resource in a per-machine location on the machine, such as a file in the Program Files folder, or a registry key or value in HKEY_LOCAL_MACHINE.

While this may not be an issue with Windows XP or earlier versions of Windows, Windows Vista and later do not give users full administrator privileges by default. Therefore, since a deferred-in-system-context action runs with user impersonation when ALLUSERS is not set, the custom action could fail.

The recommended method for preventing this behavior is to ensure that a per-machine installation is always performed by setting ALLUSERS. Per-machine installations are generally easier to manage than per-user installations.

Default Value of ALLUSERS

The ALLUSERS property is set to 1 by default. If you configure your installation so that it can be installed per user without administrative privileges, you may want to consider changing the value of the ALLUSERS property.

Task ***To configure the value of ALLUSERS:***

1. In the View List under **Organize Your Setup**, click **General Information**.
2. For the **ALLUSERS** setting, select the appropriate value.

For information about the available options, see General Information Settings.

Default Controls on the Ready to Install and Customer Information Dialogs

Use the Show All Users Option setting to specify whether you want to give end users the option of installing your product for all users or for only the current user. This setting is displayed when you click the Dialogs explorer in the Dialogs view. Available options for the Show Per-User Option setting are:

- **No**—InstallShield does not include the option that enables end users to specify how they want to install the product.
- **Yes (Only Windows 7 and Later)**—If the target system has Windows 7 or Windows Server 2008 R2, InstallShield adds buttons to the Ready to Install dialog. The buttons let end users specify how they want to install the product. If elevated privileges are required, the shield icon is included on the all-users button. If an end user selects the all-users button, the **ALLUSERS** property is set to 2, and the **MSIINSTALLPERUSER** property is set to 1. If an end user selects the per-user button, the **ALLUSERS** property is set to 1, and the **MSIINSTALLPERUSER** property is not set.
- **Yes (All Systems)**—If the target system has Windows 7 or Windows Server 2008 R2, InstallShield adds buttons to the Ready to Install dialog. The buttons let end users specify how they want to install the product. If elevated privileges are required, the shield icon is included on the all-users button. If an end user selects the all-users button, the **ALLUSERS** property is set to 2, and the **MSIINSTALLPERUSER** property is set to 1. If an end user selects the per-user button, the **ALLUSERS** property is set to 1, and the **MSIINSTALLPERUSER** property is not set.

  If the target system has Windows Vista or earlier, or Windows Server 2008 or earlier, InstallShield adds radio buttons to the Customer Information dialog. The radio buttons let end users specify how they want to install the product. If an end user selects the all-users radio button and the end user has elevated privileges, the **ALLUSERS** property is set to 1. If an end user selects the per-user button and the end user has elevated privileges, the **ALLUSERS** property is set to an empty string ("").

  The default value is No.

Customizing Installation Behavior

An important aspect of creating an installation is customizing it for your end users' needs. The help topics in "Customizing Installation Behavior" discuss various features of InstallShield that help you extend the functionality of your installation. For example, you may find it useful to create custom actions to add support for something not directly supported by Windows Installer. For more information on how you can customize the installation behavior in your project, refer to this section of the documentation.

Using Custom Actions

Windows Installer provides many standard actions that execute an installation. However, there may be times where your installation needs expanded functionality. In these cases, custom actions let you extend the capabilities of standard actions. This can be done through including dynamic-link libraries, executable files, or script in your installation. The following table describes the different types of custom actions that you can add to your projects.

Table 4-1 ▪ Types of Custom Actions

Type of Action	Description
Windows Installer DLL (MSI DLL)	This type of action is a C-callable library that exports functions with a fixed prototype. Benefits of this action type include: ● Windows Installer DLLs usually have minimal dependencies. ● Much Windows functionality is exposed in system libraries. ● Unlike executable-file actions, this type of action can use the Windows Installer API to access the running Windows Installer session and work with properties, directories, and so on. Note that writing MSI DLLs requires C/C++ and Windows API coding skills.

Table 4-1 ▪ Types of Custom Actions (cont.)

Type of Action	Description
Standard DLL	This type of action is an InstallShield extension that lets you call functions from C-callable DLLs. The type of function prototype that you select for the action (classic or new) determines which signature you must use for your function. ● **Classic**—The prototype in this type of DLL must follow the same format that was used in early versions of InstallShield Express. ● **New**—The prototype in this type of DLL uses the same signature that is required for MSI DLL custom actions. If you want to use this type of signature for a new custom action that you are adding to your project, it is recommended that you use an MSI DLL custom action instead of a standard DLL, since an MSI DLL action offers more flexible scheduling options. ***Important ▪*** *Note that for standard DLL custom actions, a non-zero return value indicates success, and a return value of zero indicates failure. For MSI DLL custom actions, a return value of zero indicates success; non-zero return values indicate statuses such as failure or cancellation.*
Executable File	This type of action launches an executable file that is included in your installation (as a temporary support file or installed with the product) or that already exists on the target system. A benefit of this action type is that a lot of operating system functionality is exposed in executable files. For example, this type of action lets your installation open text files and other common file types, set permissions on existing directories, and launch batch files. Executable-file actions do not have access to the running installation session. Thus, you cannot pass Windows Installer properties in to the executable file (except as command-line arguments) or back from the executable file (except through external storage, such as the registry or a file). Note that to open a document or launch a batch file, you must explicitly refer to an executable file that handles that type of document.

Table 4-1 ▪ Types of Custom Actions (cont.)

Type of Action	Description
VBScript or JScript	This type of action runs VBScript or JScript code. Benefits of this action type include: • Script actions are often easy to implement. • Unlike executable-file actions, this type of action can use the Windows Installer API to access the running Windows Installer session and work with properties, directories, and so on. • This type of action can be useful for simple tasks such as string manipulation. Note that VBScript and JScript actions often trigger antivirus or antispyware programs when they access or modify the target system.

Edition ▪ *The Premier and Professional editions of InstallShield include support for several custom action types that are not available in the Express edition. These extra custom action types enable you to do the following: run InstallScript code, set a property, set a directory, call a public method in a managed assembly, or display an error message under certain conditions and abort the installation.*

Windows Installer DLL Custom Actions

If you need to perform an action in your installation that is not supported by InstallShield or the Windows Installer service, you can create a custom action that calls an entry-point function from a Windows Installer .dll file. With this type of custom action, often called an *MSI DLL action*, the function must be defined with the following signature:

```
UINT __stdcall FunctionName (MSIHANDLE hInstall) {...}
```

The function name can vary; however, the return type, calling convention, and single parameter must use the types that are specified in the aforementioned signature. The MSIHANDLE data type that is used in the function signature is a handle to the running installation.

The following sample code is for a function called **MyFunctionName**.

```
UINT __stdcall MyFunctionName (MSIHANDLE hInstall)
{
    MessageBox(
        GetForegroundWindow( ),
        TEXT("This is MyFunctionName."),
        TEXT("Custom Action"),
        MB_OK | MB_ICONINFORMATION);
    return ERROR_SUCCESS;
}
```

Your entry-point function should return ERROR_SUCCESS to indicate that the action finished successfully. If your custom action returns a non-zero value and you selected No for the custom action's Ignore Exit Code setting, the installation exits. If you selected Yes for the custom action's Ignore Exit Code setting, the installation continues, regardless of the custom action's return code.

Once your MSI DLL is ready, you must design a custom action to call the entry-point function. To learn how to create the custom action, see Adding an MSI DLL Custom Action to Your Project.

Requirements

- Header: Declared in `Msiquery.h`
- Library: Use `Msi.lib`

Both `Msiquery.h` and `Msi.lib` can be found in the Windows Installer SDK, which can be downloaded from Microsoft's Web site. In addition to the two files listed above, the Windows Installer SDK contains Microsoft's definitive documentation on the Windows Installer APIs.

Adding an MSI DLL Custom Action to Your Project

The items in the Custom Actions view are organized by chronological order, according to when they are launched during installation or uninstallation. When you add a custom action to your project, you specify when the custom action should be launched by adding the action to the appropriate installation or uninstallation item.

Task ***To add an MSI DLL custom action to your project:***

1. In the View List under **Define Setup Requirements and Actions**, click **Custom Actions**.
2. In the **Custom Actions During Installation** explorer, the **Custom Actions During Maintenance** explorer, or the **Custom Actions During Uninstallation** explorer, click the item that should contain the custom action that you want to create.
3. Right-click the item and click **New MSI DLL**. InstallShield adds a new custom action with the name **NewCustomAction***n* (where *n* is a successive number).
4. Type a name for the custom action.
5. Configure the settings for the custom action.

Configuring an MSI DLL Custom Action's Settings

When you add an MSI DLL file custom action to your project, you need to configure its settings.

Task ***To configure the settings for an MSI DLL custom action in your project:***

1. In the View List under **Define Setup Requirements and Actions**, click **Custom Actions**.
2. In the **Custom Actions During Installation** explorer, the **Custom Actions During Maintenance** explorer, or the **Custom Actions During Uninstallation** explorer, click the item that should contain the custom action that you want to configure.

3. Configure the settings in the grid on the right.

DLL Custom Actions

If you need to perform an action in your installation that is not supported by InstallShield or the Windows Installer service, you can create a custom action that calls a function from a .dll file. This .dll file can perform any function that you require, such as verifying a serial number.

The first step in creating a custom action that calls a DLL function is to write the DLL. There are two function prototypes that you can use to write your custom action:

- Classic DLL custom action function prototype—The prototype in this type of DLL must follow the same format that was used in early versions of InstallShield Express.
- New DLL custom action function prototype—The prototype in this type of DLL uses the same signature that is required for MSI DLL custom actions. If you want to use this type of signature for a new custom action that you are adding to your project, it is recommended that you use an MSI DLL custom action instead of a standard DLL, since an MSI DLL action offers more flexible scheduling options.

Once your DLL is ready, you must design a custom action to call the entry-point function. To learn how to create the custom action, see Adding a DLL Custom Action to Your Project.

Classic DLL Custom Action Function Prototype

InstallShield requires a precise prototype for an entry-point function in a DLL called as the result of a custom action.

Revenera does not provide technical support for Windows programming or DLL debugging. You are responsible for correctly writing any DLL functions. Prototype your custom DLL functions as shown below. Any variation in return type or type and number of parameters can cause the custom action to fail.

```
LONG WINAPI Foo(HWND, LPTSTR , LPTSTR, LPTSTR, LPTSTR);
```

InstallShield uses the function prototype to pass the following information to your DLL:

1. Parameter 1 passes the installation's window handle. This parameter always returns NULL.
2. Parameter 2 passes the source directory [SRCDIR].
3. Parameter 3 passes the support directory [SUPPORTDIR].
4. Parameter 4 passes the main target directory [INSTALLDIR].
5. Parameter 5 passes the database directory [DATABASEDIR].

If you are prototyping a custom action to handle the serial number entered in the Customer Information run-time dialog, then Parameter 4 will be the serial number.

The body of your DLL function can do just about anything you want. Obviously, you may have good use for the values passed to the function by the installation.

Your DLL function must return a value of type LONG as a state flag signaling the completion of the routine. If your function returns zero and you selected No for the custom action's Ignore Exit Code setting, the installation exits. If it returns any other value, or if you selected Yes for the custom action's Ignore Exit Code setting, the installation continues.

Sample DLL Function

This section contains sample source code. This sample contains a function, Foo(), that displays a message box showing the values that were passed to the function.

```
#include <windows.h>
#ifdef __cplusplus
extern "C" {
#endif

// Foo() function definition.
LONG WINAPI Foo(HWND hwnd, LPSTR szSrcDir, LPSTR szSupport, LPSTR szInst, LPSTR szDbase)
  { CHAR szTmp[1024];
  int ret;

  // Construct a string to display the values passed into Foo().
  wsprintf(szTmp, "Extension called. hwnd=%x szSrcDir=%s szSupport=%s szInst=%s. Do you want" \
            "to exit now?", hwnd, szSrcDir, szSupport, szInst);

  // Display the string in a message box.
  ret=MessageBox(hwnd, szTmp, "Test Extension", MB_YESNO);
  if (ret==IDYES)
    // Returning 0 causes the installation to end.
    return(0);
  else
    // Returning non-zero causes the installation to continue.
    return(1);
}

#ifdef __cplusplus
}
#endif
```

When a non-zero value is returned by your function, the installation continues. If zero is returned, the installation exits.

To be able to call the above DLL function, you must also include a definition (.def) file when you build the DLL to export the function properly. Include the following definition file with your project. The name after LIBRARY should be the name you have given your DLL.

```
; mydll.def : Declares the module parameters for the DLL.

LIBRARY MYDLL
DESCRIPTION 'sample Windows Dynamic Link Library'

EXPORTS
    Foo @1
```

New DLL Custom Action Function Prototype

The new function signature enables you to get a handle to the .msi database that is currently running. Once you have a handle to the database, you can call any number of Windows Installer APIs. The example below shows how to retrieve the value of the ProductName property.

```
UINT __stdcall Action(MSIHANDLE hInstall)
  {
```

```
    TCHAR buffer[32] = {0};
    DWORD dWord = 32;
    MsiGetProperty(hInstall, TEXT("ProductName"), buffer, &dWord);
    MessageBox(0,buffer,TEXT("Showing Product Name"), MB_OK);
    return 1;
}
```

Your function should return a non-zero value to indicate that the action finished successfully and that the installation should continue. If your function returns the number 0 and you selected No for the custom action's Ignore Exit Code setting, the installation exits. If you selected Yes for the custom action's Ignore Exit Code setting, the installation continues, regardless of the custom action's return code.

Requirements

- Header: Declared in `Msiquery.h`
- Library: Use `Msi.lib`

Both `Msiquery.h` and `Msi.lib` can be found in the Windows Installer SDK, which can be downloaded from Microsoft's Web site. In addition to the two files listed above, the Windows Installer SDK contains Microsoft's definitive documentation on the Windows Installer APIs.

Adding a DLL Custom Action to Your Project

The items in the Custom Actions view are organized by chronological order, according to when they are launched during installation or uninstallation. When you add a custom action to your project, you specify when the custom action should be launched by adding the action to the appropriate installation or uninstallation item.

Task ***To add a DLL custom action to your project:***

1. In the View List under **Define Setup Requirements and Actions**, click **Custom Actions**.
2. In the **Custom Actions During Installation** explorer, the **Custom Actions During Maintenance** explorer, or the **Custom Actions During Uninstallation** explorer, click the item that should contain the custom action that you want to create.
3. Right-click the item and click **New DLL**. InstallShield adds a new custom action with the name **NewCustomAction***n* (where *n* is a successive number).
4. Type a name for the custom action.
5. Configure the settings for the custom action.

Configuring a DLL Custom Action's Settings

When you add a DLL file custom action to your project, you need to configure its settings.

Task ***To configure the settings for a DLL custom action in your project:***

1. In the View List under **Define Setup Requirements and Actions**, click **Custom Actions**.
2. In the **Custom Actions During Installation** explorer, the **Custom Actions During Maintenance** explorer, or the **Custom Actions During Uninstallation** explorer, click the item that should contain the custom action that you want to configure.
3. Configure the settings in the grid on the right.

Executable File Custom Actions

You may need to launch a third-party installation because you cannot repackage another vendor's installation. Or, maybe you want to play an .avi while the installation runs. If that is the case, you can create a custom action that launches an executable file.

One of the supported custom action types is launching an executable in the installation. To learn how to create this type of custom action, see Adding an .exe Custom Action to Your Project.

Adding an .exe Custom Action to Your Project

The items in the Custom Actions view are organized by chronological order, according to when they are launched during installation or uninstallation. When you add a custom action to your project, you specify when the custom action should be launched by adding the action to the appropriate installation or uninstallation item.

Task ***To add an .exe custom action to your project:***

1. In the View List under **Define Setup Requirements and Actions**, click **Custom Actions**.
2. In the **Custom Actions During Installation** explorer, the **Custom Actions During Maintenance** explorer, or the **Custom Actions During Uninstallation** explorer, click the item that should contain the custom action that you want to create.
3. Right-click the item and click **New EXE**. InstallShield adds a new custom action with the name **NewCustomAction***n* (where *n* is a successive number).
4. Type a name for the custom action.
5. Configure the settings for the custom action.

Configuring an .exe Custom Action's Settings

When you add an .exe file custom action to your project, you need to configure its settings.

Task ***To configure the settings for an .exe custom action in your project:***

1. In the View List under **Define Setup Requirements and Actions**, click **Custom Actions**.
2. In the **Custom Actions During Installation** explorer, the **Custom Actions During Maintenance** explorer, or the **Custom Actions During Uninstallation** explorer, click the custom action that you want to configure.
3. Configure the settings in the grid on the right.

VBScript and JScript Custom Actions

Depending on your installation's requirements, you may need to create a custom action that calls an existing VBScript file (.vbs) or JScript (.js). To learn how to create this type of custom action, see Adding a VBScript or JScript Custom Action to Your Project.

VBScript Custom Action Example

The following is an example of a VBScript custom action that determines the value of INSTALLDIR and changes it depending on the presence of a file.

```
' Get the value of INSTALLDIR
Dim sInstalldir
sInstalldir = Session.Property("INSTALLDIR")

' Show it.
MsgBox sInstalldir

' Check to see if a file exists
Dim pFs
Set pFs = CreateObject("Scripting.FileSystemObject")
Dim sSomeFile
sSomeFile = sInstalldir & "31337.txt"
If pFs.FileExists(sSomeFile) Then
  ' If it exists, change INSTALLDIR
  Session.Property("INSTALLDIR") = sInstalldir & "New"
End If
```

Adding a VBScript or JScript Custom Action to Your Project

The items in the Custom Actions view are organized by chronological order, according to when they are launched during installation, maintenance, or uninstallation. When you add a custom action to your project, you specify when the custom action should be launched by adding the action to the appropriate installation, maintenance, or uninstallation item.

Task ***To add a VBScript or JScript custom action to your project:***

1. In the View List under **Define Setup Requirements and Actions**, click **Custom Actions**.
2. In the **Custom Actions During Installation** explorer, the **Custom Actions During Maintenance** explorer, or the **Custom Actions During Uninstallation** explorer, click the item that should contain the custom action that you want to create.
3. Right-click the item and click **New VBScript** or **New JScript**. InstallShield adds a new custom action with the name **NewCustomAction***n* (where *n* is a successive number).
4. Type a name for the custom action.
5. Configure the settings for the custom action.

Configuring a VBScript or JScript Custom Action's Settings

When you add a VBScript or JScript custom action to your project, you need to configure its settings.

Task ***To configure the settings for a VBScript or JScript custom action in your project:***

1. In the View List under **Define Setup Requirements and Actions**, click **Custom Actions**.
2. In the **Custom Actions during Installation** explorer or the **Custom Actions during Uninstallation** explorer, click the custom action that you want to configure.
3. Configure the settings in the grid on the right.

Action Execution Options

Custom actions are executed in the order in which they appear in a sequence. Some actions need to be scheduled to run immediately at the start of the installation; others need to be deferred for a later time. The In-Script Execution setting in the Custom Actions view enables you to select which iteration of the sequence (deferred, rollback, or commit) should trigger your action. This setting also lets you specify the context for deferred, rollback, and commit actions: actions can run in user context (with the privileges of the user running the installation) or in system context (with elevated privileges). If this setting is not displayed for a particular custom action in the Custom Actions view, that action is scheduled for immediate execution; immediate-execution custom actions always run in user context.

Immediate Execution

As its name implies, immediate-execution custom actions runs when the internal Windows Installer installation script is being compiled. When an .msi file is launched, the Windows Installer service converts all the tables in the installation database into an internal script. This script is built by cycling through all the actions in the installation in the order in which they appear. The building of this script is immediate execution. When an action that is set for immediate execution is encountered, that action is executed. Therefore, this action launches before any file transfers are encountered, possibly even before the end-user interface for the installation is fully loaded.

As a rule, custom actions that are scheduled for immediate execution do not modify the target system, but only set properties and query the target system (for example, to see if the target system meets your product's system requirements). Custom actions that set Windows Installer properties must be scheduled for immediate execution, and custom actions that occur in the User Interface sequence must be scheduled for immediate execution.

Because actions of this type run before any changes have been made to the system, they cannot rely on files that are being installed by the installation.

Deferred Execution

Deferred execution takes place when the internal script that is generated during immediate execution is executed by the Windows Installer. After that script has been fully generated, the Windows Installer runs the newly compiled script. The script runs through all of the actions in your sequences and executes them in order. However, if an action is scheduled for immediate execution, the action does not run again during deferred execution.

Actions that launch during deferred execution have access to files that are being installed as part of the installation. As a result, you can call a custom action that calls a function from a DLL file that is installed with your product during this phase of the installation. However, deferred execution custom actions must take place between InstallInitialize and InstallFinalize in order to work properly.

Custom actions that rely on a file that is being installed to the target system, or that rely on other system changes to have already been performed, must be scheduled for deferred execution.

Rollback Execution

Rollback occurs when an error is encountered by the installation or the end user cancels the installation before it has a chance to complete. The rollback execution option allows you to set your action to execute only during rollback. Therefore, any actions that are enabled for rollback execution are written to the installation script, as are deferred execution actions. Unlike deferred execution actions, rollback execution actions launch only during rollback. (Rollback custom actions run only if the installation fails during deferred execution.)

Any custom actions that make changes to the target system during an installation should be undone with a rollback execution custom action in the event of rollback. For example, if you have a custom action that creates a file, you should create a second custom action that deletes the file, and schedule the second action for rollback execution. (A rollback custom action should be scheduled before the custom action it reverses.)

Commit Execution

Commit execution actions do not run until the InstallFinalize action completes successfully. This means that they wait until the installation has completed transferring files, registering COM servers, and creating shortcuts and registry entries. Then, any actions that are set to commit execution launch in the order in which they appear in the sequence.

For example, if you have a custom action that creates a temporary file, you should create a second custom action that deletes the file, and schedule it for commit execution.

Deferred, Rollback, or Commit Execution in System Context

Like deferred execution actions, deferred-execution-in-system-context actions do not launch until the script that generated by the Windows Installer is run. However, actions of this type execute with no user impersonation.

Similarly, rollback-execution-in-system-context and commit-execution-in-system-context actions execute with no user impersonation.

Because of the elevation of privileges on Windows Vista and later systems, deferred custom actions must be made in system context if they need to make system changes that require elevated privileges. Deferred custom actions in user context may not have elevated privileges even if an administrator runs the installation on a Windows Vista or later system, and thus should be avoided for custom actions requiring higher-level privileges.

Note ▪ *Custom actions that run in the system context generally do not have access to network resources.*

Changing When Custom Actions Are Launched

The items in the Custom Actions view are organized by chronological order, according to when they are launched during installation, maintenance, or uninstallation. When you add a custom action to your project, you specify when the custom action should be launched by adding the action to the appropriate installation, maintenance, or uninstallation item.

Task ***To change when a custom action is launched, do one of the following:***

- Drag the custom action from its current location to the new location.
- Right-click the custom action and click **Move Up** or **Move Down**.
- Click the custom action and then press CTRL+SHIFT+UP ARROW or CTRL+SHIFT+DOWN ARROW.

The Custom Actions view is divided into the following main run-time categories:

- **Custom Actions During Installation**—Actions that are scheduled during this run-time category are launched during a first-time installation of the product.
- **Custom Actions During Maintenance**—Actions that are scheduled during this run-time category are launched during modify mode and repair mode; they are also launched when the product is silently uninstalled.
- **Custom Actions During Uninstallation**—Actions that are scheduled during this run-time category are launched when the product is being uninstalled.

Using a Custom Action for Serial Number Validation

Serial number validation requires the use of a custom DLL file. Example code for this type of DLL file is available in the following location:

InstallShield Program Files Folder\Samples\WindowsInstaller\ValidateSerialNumber

In this example, the serial number must take the following format:

```
Field1-1505-XXXXXXXXXX
```

The first section must appear as it does above, although it is not case-sensitive. The second section, 1505, must be a number evenly divisible by 5 (for example, 1111 would not work). The last section can be any 10 alphanumeric characters. Keep in mind that this format is merely an example. Your serial number function can take any format that you would like it to have.

Filter the Input Characters

To filter the input from illegal characters, enter **??????-####-??????????** in the Serial Number Template field for the Customer Information dialog in the Dialogs view. The question marks (?) signify alphanumeric characters and the number signs (#) signify numbers. By setting the template in this manner, your serial number field is separated into three parts, and each section is filtered for different types of input.

Further Considerations

The Validate Function, Success Return Value, and Retry Limit properties allow you to further customize how your serial number validation DLL file works. For the Validate Function property, enter the name of the function within your .dll file that validates the serial number provided by the end user.

Your .dll file should return a specific value every time it is run. The example .dll included with InstallShield returns 1 on success and –1 on failure. Therefore, type **1** in the Success Return Value field. When you write your own .dll file, the success return value can be whatever you choose, as long as it is non-zero.

The Retry Limit property lets you set how many failed attempts are allowed. For example, if you only want users to be able to try entering a serial number three times, type **3** in this field. After the third failed attempt, the installation exits.

The example .dll file contains a user interface element that breaks down the entered serial number and displays it to the user. Additionally, it displays the section, if any, of the serial number that is incorrect. This display is useful when testing and troubleshooting your serial number verification .dll file, but it should not be left in the final version of your installation.

Note ▪ *InstallShield sequences the serial number DLL file custom action after any user-defined custom actions. Thus, if you add your own custom action and schedule it after the Customer Information dialog, the installation launches this custom action before the serial number DLL file custom action.*

Custom Action Gallery

The Custom Action gallery contains preconfigured custom actions that achieve common functionality. The custom actions contained in this gallery do not require any input from you. They are designed to be added to your installation project and run during installation. To insert an action from your gallery into your installation, right-click one of the items in the Custom Actions view, point to Add from Gallery, and then click the custom action that you want to add to your project.

Currently, the only action listed in this gallery is Schedule Reboot at End action. If you add this action to your project, it is automatically added to the After Setup Complete Success dialog part of the installation. This action forces the target machine to reboot after the installation completes, allowing .dll files and other files to register with the system at startup. Changing any of this custom action's settings or scheduling it for a different part of the installation is not recommended. For example, if you reschedule it, your installation may cause the target machine to restart before the file transfer has completed.

Checking if a Product Is Installed

The following code will check if a product with a ProductCode of {8FC71000-88A0-4B41-82B8-8905D4AA904C} is installed on the machine. This code can be used as a New Style DLL Custom Action.

```
UINT __stdcall CheckProduct(MSIHANDLE hInstall)
{
   int RetVal = 0;
   RetVal = MsiQueryProductState("{8FC71000-88A0-4B41-82B8-8905D4AA904C}");
      if (RetVal==5))
      {
         MessageBox(GetForegroundWindow(),TEXT("Installed"),TEXT("My Product"), MB_OK);
         return 1;
      }
      else
      {
         MessageBox(GetForegroundWindow(),TEXT("Not Installed"),TEXT("My Product"), MB_OK);
         return 0;
      }
}
```

The following table lists the return values of MsiQueryProductState:

Table 4-2 ▪ Return Values of MsiQueryProductState

Return Value	Description
–1	The product is neither advertised nor installed.
1	The product is advertised but not installed.
2	The product is installed for a different user.
5	The product is installed for the current user.

Using Setup Files

Setup files (also known as support files) are files that are available on the target system only during your application's installation process. Support files are copied to a temporary directory on the target system when installation begins, and they are deleted when the installation is complete. The support directory (SUPPORTDIR) is a dynamic file location and might be different on every target system and even on the same system for different installation instances.

In the Setup Files view, you can add and remove files that you want to be available on the target system only during installation.

To access a particular setup file during installation, you can query for the value of the support directory (SUPPORTDIR) and then append the file name to the SUPPORTDIR value to obtain the complete path of the file.

Adding Setup Files

Task ***To add setup files to your installation project:***

1. In the View List under **Define Setup Requirements and Actions**, click **Setup Files**.
2. Optionally, to create one or more subfolders for your files, in the **Support Files** explorer, right-click the English (United States) area (or to another language area if your project is in a language other than English), and then click **New Folder**. InstallShield adds a folder. Enter the name that you want to use for the folder.
3. In the **Support Files** explorer, click the item that should contain the support file that you are adding. For example, you can add the setup file to the English (United States) item or an item for a different language if your project is in a language other than English.
4. Right-click anywhere in the **Files** pane and then click **Insert Files**. The **Open** dialog box opens.
5. Browse to the file that you want to include. To select multiple files, hold down the CTRL key while clicking files.
6. Click **OK**.

InstallShield adds the file to the Files pane.

Adding a License File

You can add a text file containing a license agreement in the Support Files view (for example, License.txt).

Task ***To add a license file to your installation project:***

1. In the View List under **Define Setup Requirements and Actions**, click **Setup Files**.
2. Optionally, to create one or more subfolders for your license, in the **Support Files** explorer, right-click the English (United States) area (or to another language area if your project is in a language other than English), and then click **New Folder**. InstallShield adds a folder. Enter the name that you want to use for the folder.
3. In the **Support Files** explorer, click the item that should contain the license file. For example, you can add the license file to the English (United States) item or an item for a different language if your project is in a language other than English.
4. Right-click anywhere in the **Files** pane and then click **Insert Files**. The **Open** dialog box opens.
5. Browse to the license file that you want to include. To select multiple files, hold down the CTRL key while clicking files.
6. Click **OK**.

InstallShield adds the license file to the Files pane.

Accessing a Setup File During Installation

To access a particular setup file during installation, you can query for the value of the support directory (SUPPORTDIR) and then append the file name to the SUPPORTDIR value to get the complete path of the file. The following VBScript, when run as a custom action, provides the location for SUPPORTDIR:

```
MsgBox (Session.Property("SUPPORTDIR"))
```

Sorting Support Files

You can sort the files and folders in the Files pane of the Setup Files view.

Task ***To sort files and folders in the Setup Files view:***

1. In the **Support Files** explorer, click the item that contains the support files and folders that you would like to sort.
2. In the **Files** pane, click the heading of a column to sort by that column.

Disk1 Files

The Disk1 feature in the Setup Files view enables you to add files and folders on Disk1 of the installation media; these files and folders are not automatically installed to the target system when your installation is run. You can link to the installation media from your application or from the installation program.

For example, you might include a large redistributable file with your application that you want end users to be able to access, but do not want to include in the application installation. This file should be placed in the Disk1 folder.

Disk1 files are taken into account when InstallShield calculates the size of the media; therefore, InstallShield ensures that the addition of a Disk1 file will not cause the build to exceed the disk size specification for that release. This feature is useful if you want to store a redistributable on your disk image so that your installation can launch it from the source media at installation time.

Adding Disk1 Files

Task ***To add Disk1 files to your installation project:***

1. In the View List under **Define Setup Requirements and Actions**, click **Setup Files**.
2. In the **Support Files** explorer, click **Disk1**.
3. Right-click anywhere in the **Files** pane and then click **Insert Files**. The **Open** dialog opens.
4. Browse to the file that you want to add to Disk1. To select multiple files, hold down the CTRL key while clicking files.
5. Click **OK**.

InstallShield adds the file to the Files pane.

Adding Disk1 Folders

You can add an entire folder, plus all of its contents, to your Disk1 folder. Then when you build your project, InstallShield adds the folder and its files to the root of your installation media.

Task ***To add Disk1 folders to your installation project:***

1. In the View List under **Define Setup Requirements and Actions**, click **Setup Files**.
2. In the **Support Files** explorer, click **Disk1**.
3. Right-click anywhere in the **Files** pane and then click **Insert Folder**. The **Browse for Folder** dialog opens.
4. Browse to the folder that you want to add to Disk1.
5. Click **OK**.

InstallShield adds the folder to the Files pane.

Removing Disk1 Files and Folders

Task ***To remove a file from Disk1:***

1. In the View List under **Define Setup Requirements and Actions**, click **Setup Files**.
2. In the **Support Files** explorer, click **Disk1**.
3. In the **Files** pane, right-click the file or folder that you want to remove and then click **Delete**.

Removing Setup Files

Task ***To remove setup files from your project:***

1. In the View List under **Define Setup Requirements and Actions**, click **Setup Files**.
2. In the **Support Files** explorer, click the language-specific item that contains the support file that you want to remove.
3. In the **Files** pane, right-click the file and click **Delete**.

Configuring Servers

When you are creating an installation, you may find it necessary to provide server-side support for some technology that will be installed on a target system. InstallShield makes it easy to configure server-side installations. You can manage new Internet Information Services (IIS) Web sites and manage COM+ applications and components with installations that you create in InstallShield.

Managing COM+ Applications and Components

With the Component Services view in InstallShield, you can manage COM+ server applications and components for your installation package.

Note the following information regarding component services in InstallShield:

- Only non-system COM+ applications can be added to your project. Therefore, InstallShield displays only non-system COM+ applications under the COM+ Applications explorer in the Component Services view.
- Only the COM+ applications that are installed on the local machine are included in the Component Services view and available for you to add to your projects.

The look and feel of the Component Services view is similar to that of the Component Services administrative tool in the Control Panel.

***Edition** • InstallShield Premier Edition and InstallShield Professional Edition provide additional functionality in the Component Services view; these editions enable you to manage both COM+ server applications and application proxies. A COM+ application proxy consists of a subset of the attributes of the server application, and it enables remote access from a client machine to the machine where the application resides.*

Adding COM+ Applications

Task ***To add a COM+ application to your installation:***

1. Open the **Component Services** view.
2. Under the **COM+ Applications** item, select the check box for the application that you would like to add to your project. The tabbed property sheets open.
3. On the **Installation** property sheet, set the parameters for server installations. This property sheet is also where you select the features to which the COM+ application belongs.
4. Use the other tabbed property sheets to set standard COM+ application properties. These property sheets correspond to the tabs on the Properties dialog box associated with each COM+ application when administered from the Component Services administrative tool in the Control Panel.

Removing COM+ Applications

Task ***To remove a COM+ application from your installation:***

1. Open the **Component Services** view.
2. Under the **COM+ Applications** item, clear the check box for the application that you would like to remove from your project.

Managing Internet Information Services

Internet Information Services (IIS) is a Web server developed by Microsoft. It provides a secure platform for building and deploying Web-based applications, managing Web sites, and publishing information to the Internet or an intranet.

The Internet Information Services view in InstallShield enables you to create and manage a new IIS Web site, applications, and virtual directories.

Edition ▪ *The Express edition of InstallShield includes support for installing only one Web site per installation.*

The following editions of InstallShield support the creation of more than one Web site per installation. These editions also include support for managing IIS application pools and Web service extensions:

- *Premier*
- *Professional*

In addition, the Premier edition of InstallShield includes an IIS scanner that checks an existing IIS Web site and records data about the settings for the Web site, its virtual directories, its applications, and its application pools. You can import that IIS data into the Internet Information Services view in InstallShield Premier Edition. Once you have imported the IIS data into a project, you can use the Internet Information Services view to make changes to the IIS settings as needed.

Version-Specific Information for IIS Support in InstallShield

Edition ▪ *The Express edition of InstallShield includes support for installing only one Web site per installation.*

The following editions of InstallShield support the creation of more than one Web site per installation. These editions also include support for managing IIS application pools and Web service extensions:

- *Premier*
- *Professional*

In addition, the Premier edition of InstallShield includes an IIS scanner that checks an existing IIS Web site and records data about the settings for the Web site, its virtual directories, its applications, and its application pools. You can import that IIS data into the Internet Information Services view in InstallShield Premier Edition. Once you have imported the IIS data into a project, you can use the Internet Information Services view to make changes to the IIS settings as needed.

Following is information about specific versions of IIS:

- IIS is included with Windows 2000 Server and later and Windows XP and later systems. IIS 6 is available only on Windows Server 2003 systems. IIS 7 is available on Windows Vista and Windows Server 2008 systems. IIS 7.5 is available on Windows 7 and Windows Server 2008 R2 systems. IIS is not installed automatically by default.
- Some of the Web site and virtual directory settings in the Internet Information Services view apply to specific versions of IIS. Version-specific information is noted for these settings in the inline help panes in InstallShield. If you configure a version-specific property but a target system does not have the corresponding version of IIS, IIS ignores the version-specific property.

  For example, IIS 7 and IIS 6 do not support the Application Protection property for applications or virtual directories. In the Internet Information Services view, this property is configured through a setting in the Application Settings area for an application or a virtual directory. When you select this setting in the Internet Information Services view, the help pane that is displayed in the lower-right corner indicates that the setting does not apply to IIS 6 or later. If you configure this setting and an end user installs your product on a target system that has IIS 6 or later, the Application Protection setting is ignored.
- Windows Vista and later and Windows Server 2008 and later systems store settings for individual Web sites, applications, and virtual directories in configuration files that are located at the physical path that you specify in your installation project—in the Content Source Path (Local or UNC) setting in the Internet Information Services view. Therefore, each Web site, application, or virtual directory should have a unique physical path. Otherwise, you may notice unexpected behavior, especially in testing environments, if you have two different applications or virtual directories with the same physical path.

  For example, if you have two virtual directories that have the same physical path, and directory browsing is enabled in one but not the other, the directory browsing setting for the second virtual directory that is created overrides the setting for the first virtual directory.
- On systems that have Windows Server 2003, if IIS 6 is not installed, other IIS directories and sites are still created. IIS 6–specific settings are skipped.
- IIS 5.1 for Windows XP Professional can service only one Web site at a time. This is a limitation of IIS 5.1.
- InstallShield supports version 5 and later of IIS.

Run-Time Requirements for IIS Support

Edition ▪ *The Express edition of InstallShield includes support for installing only one Web site per installation.*

The following editions of InstallShield support the creation of more than one Web site per installation. These editions also include support for managing IIS application pools and Web service extensions:

- *Premier*
- *Professional*

In addition, the Premier edition of InstallShield includes an IIS scanner that checks an existing IIS Web site and records data about the settings for the Web site, its virtual directories, its applications, and its application pools. You can import that IIS data into the Internet Information Services view in InstallShield Premier Edition. Once you have imported the IIS data into a project, you can use the Internet Information Services view to make changes to the IIS settings as needed.

IIS support in InstallShield installations works only if IIS is installed on the target machine and the end user has administrative privileges.

During the installation of a package that includes IIS settings, an InstallShield installation checks for the existence of IIS on the target machine. If IIS is not installed, the installation displays a dialog informing the end user that they do not have IIS installed. The dialog gives the end user the option to abort, retry, or ignore:

- If the end user chooses to abort, the installation stops.
- If the end user installs IIS and then chooses to retry, the installation checks for IIS again, and continues with the installation. If the end user does not install IIS but still chooses to retry, the installation checks for IIS again and then displays the dialog again.
- If the end user chooses to ignore, the installation continues, but the IIS Web site and any virtual directories are not configured.

***Note** ▪ InstallShield does not support the creation of Web sites on target machines other than the one on which the installation is running.*

Specifying Whether a Web Server Should Allow the CMD Command to Be Used for SSI #exec Directives

***Edition** ▪ The Express edition of InstallShield includes support for installing only one Web site per installation.*

The following editions of InstallShield support the creation of more than one Web site per installation. These editions also include support for managing IIS application pools and Web service extensions:

- *Premier*
- *Professional*

In addition, the Premier edition of InstallShield includes an IIS scanner that checks an existing IIS Web site and records data about the settings for the Web site, its virtual directories, its applications, and its application pools. You can import that IIS data into the Internet Information Services view in InstallShield Premier Edition. Once you have imported the IIS data into a project, you can use the Internet Information Services view to make changes to the IIS settings as needed.

Server-side include (SSI) directives instruct a Web server to insert content into a Web page. The #exec type of directive enables the Web server to include the output of a shell command in a Web page.

You can configure an IIS Web server to prevent the CMD command for the #exec directive from being used to execute shell commands, or you can configure it to allow the CMD command to be used to execute this type of command. The SSIEnableCmdDirective registry value for the HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W3SVC\Parameters registry key is what determines whether the CMD command is permitted.

InstallShield lets you specify how your installation should configure the SSIEnableCmdDirective registry value on target systems. If you do not want your installation to change the SSIEnableCmdDirective registry value, you can also specify that.

Because of security concerns, the default SSIEnableCmdDirective registry value is FALSE (0); the FALSE (0) value prevents end users from running unauthorized server-side executable files.

Task ***To specify whether a Web server should allow the CMD command to be used for SSI #exec directives:***

1. In the View List under **Configure the Target System**, click **Internet Information Services**.
2. In the center pane, click the **Web Sites** explorer. InstallShield displays the Web server settings in the right pane.
3. For the **SSIEnableCmdDirective registry value** setting, select the appropriate option:
   - **Ignore**—Do not change the SSIEnableCmdDirective registry value on the target system. This is the default option.
   - **FALSE (0)**—Set the SSIEnableCmdDirective registry value on the target system to 0. This prevents the #exec CMD directive of server-side includes to be used to execute shell commands. Note that if you select this value and an IIS Web server has applications that rely on #exec CMD directives, those applications may stop working properly after your installation project's Web site and virtual directory are installed.
   - **TRUE (1)**—Set the SSIEnableCmdDirective registry value on the target system to 1. This allows the #exec CMD directive of server-side includes to be used to execute shell commands.

If you select the FALSE or TRUE options, InstallShield stores the value—either 0 for FALSE or 1 for TRUE—in the `INSTALLSHIELD_SSI_PROP` property.

If a Web site or virtual directory in your installation is installed on a target system and you selected the FALSE or TRUE options for the **SSIEnableCmdDirective registry value** setting, the SSIEnableCmdDirective registry value is updated on the target system.

Note ▪ *If your product is uninstalled from a target system, the SSIEnableCmdDirective registry value is not changed, even if its value was changed during installation.*

Creating a Web Site and Adding an Application or a Virtual Directory

Edition ▪ *The Express edition of InstallShield includes support for installing only one Web site per installation.*

The following editions of InstallShield support the creation of more than one Web site per installation. These editions also include support for managing IIS application pools and Web service extensions:

- *Premier*
- *Professional*

In addition, the Premier edition of InstallShield includes an IIS scanner that checks an existing IIS Web site and records data about the settings for the Web site, its virtual directories, its applications, and its application pools. You can import that IIS data into the Internet Information Services view in InstallShield Premier Edition. Once you have imported the IIS data into a project, you can use the Internet Information Services view to make changes to the IIS settings as needed.

The Internet Information Services view in InstallShield is where you add an IIS Web site to your project. It is also where you can add applications and virtual directories to a Web site.

Task ***To create a Web site on the target system at run time:***

1. In the View List under **Configure the Target System**, click **Internet Information Services**.
2. Right-click the **Web Sites** explorer and click **Add Web Site**. InstallShield adds a new Web site.
3. Select the Web site to configure its settings.

Tip ▪ *InstallShield lets you specify the TCP port and site numbers for a Web site in your project. These settings help determine whether a new Web site is created or an existing one is updated at run time. To learn more, see Configuring the TCP Port and Site Numbers.*

Task ***To create an application on the target system at run time:***

1. In the View List under **Configure the Target System**, click **Internet Information Services**.
2. In the **Web Sites** explorer, right-click the Web site that should contain the application and click **New Application**. InstallShield adds a new application.
3. Select the application to configure its settings.

Task ***To create a virtual directory on the target system at run time:***

1. In the View List under **Configure the Target System**, click **Internet Information Services**.
2. In the **Web Sites** explorer, right-click the Web site that should contain the virtual directory and click **New Virtual Directory**. InstallShield adds a new virtual directory.
3. Select the virtual directory to configure its settings.

Tip ▪ *To learn how Web sites, applications, and virtual directories are associated with features, see Feature Associations for IIS Support.*

Creating a Nested Virtual Directory

Edition ▪ *The Express edition of InstallShield includes support for installing only one Web site per installation.*

The following editions of InstallShield support the creation of more than one Web site per installation. These editions also include support for managing IIS application pools and Web service extensions:

- *Premier*
- *Professional*

In addition, the Premier edition of InstallShield includes an IIS scanner that checks an existing IIS Web site and records data about the settings for the Web site, its virtual directories, its applications, and its application pools.

You can import that IIS data into the Internet Information Services view in InstallShield Premier Edition. Once you have imported the IIS data into a project, you can use the Internet Information Services view to make changes to the IIS settings as needed.

You can create a virtual subdirectory under an existing virtual directory.

You can also create a virtual subdirectory under a virtual directory that is being installed as part of your installation. The parent virtual directory must be installed before the virtual subdirectory.

Task ***To create a virtual directory under an existing virtual directory:***

1. In the View List under **Configure the Target System**, click **Internet Information Services**.
2. In the **Web Sites** explorer, select the Web site that should contain the nested virtual directory.
3. Right-click the new Web site and click **New Virtual Directory**. InstallShield adds a new virtual directory.
4. Click the **General** tab.
5. In the **Name** setting, indicate the name of the existing directory, as well as the name of the nested virtual subdirectory that you want to create. Separate both names with a slash.

   For example, to create a virtual directory called **MySubDirectory** under the existing virtual directory called **VirtualDirectory**, enter the following:

   ```
   VirtualDirectory/MySubDirectory
   ```

Note ▪ *If the parent directory does not already exist on the target system, the target system displays an error when the end user opens the directory in the IIS Manager.*

Configuring the TCP Port and Site Numbers

InstallShield lets you specify the TCP port and site numbers for a Web site in your project. These settings help determine whether a new Web site is created or an existing one is updated at run time. They also affect whether the Web site settings that are configured in the Internet Information Services view are applied to a Web site on the target system.

Task ***To specify the TCP port and site numbers for a Web site:***

1. In the View List under **Configure the Target System**, click **Internet Information Services**.
2. In the **Web Sites** explorer, select the Web site that you want to configure.
3. Click the **Web Site** tab.
4. In the **TCP Port** and **Site Number** boxes, enter the appropriate numbers.

Run-Time Behavior

At run time, if the Web site does not exist on the target system, the installation creates it according to the following rules:

Table 4-1 ▪ Run-Time Results for Various Sample TCP Port Number and Site Number Setting Values

TCP Port Number Setting in InstallShield	Site Number Setting in InstallShield	Result at Run Time
0	Any non-zero number	Since the TCP Port Number setting is 0 but the Site Number setting is not 0, the installation installs the Web site's applications and virtual directories to the first site number on the system. The specified site number is ignored. For example, if the first site number on the target system is 1, the installation installs the Web site's applications and virtual directories under site number 1, even if a different non-zero number such as 3 is specified for the Site Number setting. The installation does not apply any of the Web site settings that are configured in the Internet Information Services view to the Web site on the target system.
80 (a non-zero number)	0 (the default value)	If the specified TCP port exists on the target system, the installation installs the applications and virtual directories to the Web site that is running on the TCP port—in this example, port 80. The installation does not apply any of the Web site settings that are configured in the Internet Information Services view to the Web site on the target system. If the TCP port does not exist on the target system, a new Web site is created with the Web site settings that are configured in the project. In addition, the installation installs the Web site's applications and virtual directories.

Table 4-1 ▪ Run-Time Results for Various Sample TCP Port Number and Site Number Setting Values (cont.)

TCP Port Number Setting in InstallShield	Site Number Setting in InstallShield	Result at Run Time
81 (a non-zero number)	3 (a non-zero number)	If the specified TCP port and site number exist on the target system, the installation installs the applications and virtual directories under the Web site whose site number matches the specified one—in this example, site number 3. The installation does not apply any of the Web site settings that are configured in the Internet Information Services view to the Web site on the target system. If the TCP port exists on the target system but the site number does not, the installation installs the new Web site, plus its applications and virtual directories, to the existing port with the new site number. In addition, the installation configures the Web site's properties that are set in the Internet Information Services view. If the TCP port does not exist on the target system but the site number does, the installation installs the new Web site, plus its applications and virtual directories, to this TCP port. In addition, the installation configures the Web site's properties that are set in the Internet Information Services view.

Specifying the IIS Host Header Name for a Web Site

InstallShield lets you specify the host header name to identify an IIS Web site that is installed during your installation. Host headers (also known as domain names) enable you to assign more than one Web site to an IP address on a Web server.

Task ***To specify a host header name for a Web site:***

1. In the View List under **Configure the Target System**, click **Internet Information Services**.
2. In the **Web Sites** explorer, select the Web site whose host header name you want to specify.
3. Click the **Web Site** tab.
4. For the **Host Header Name** setting, type the host header name that you want to use. For example:

   ```
   www.mycompany.com
   ```

Specifying the SSL Certificate for a Web Site

A server certificate enables users to authenticate your Web server, check the validity of the Web content, and establish a secure connection. InstallShield lets you include a server certificate for a Web site in your installation so that it can be installed at run time.

Task ***To specify a SSL certificate that should be installed for a Web site:***

1. In the View List under **Configure the Target System**, click **Internet Information Services**.
2. In the **Web Sites** explorer, select the Web site whose SSL certificate you want to specify.
3. For the **SSL Certificate** setting, click the ellipsis button (**...**). The **Open** dialog box opens.
4. Select the security certificate file (.cer or .pfx) that you want to be installed, and then click **Open**.
5. If the certificate requires a password, specify it for the **SSL Certificate Password** setting.

InstallShield stores the .cer file in the `Binary` table. At run time, when the installation installs the Web site and its virtual directories, it also installs the SSL certificate.

Adding Files to an IIS Virtual Directory

Task ***To add a file to an IIS virtual directory:***

1. Add an IIS Web site to your project if you have not already done so. InstallShield automatically adds the predefined path **[IISROOTFOLDER]** to the **Files** view.
2. In the View List under **Specify Application Data**, click **Files**.
3. In the **Feature** list, select the feature with which you want the file associated.
4. In the **Destination computer's folders** pane, add the file to the **[IISROOTFOLDER]** folder, or a subfolder of the **[IISROOTFOLDER]** folder.
5. In the View List under **Configure the Target System**, click **Internet Information Services**.
6. Create a new virtual directory.
7. In the **Web Sites** explorer, click the virtual directory that you created.
8. In the **Content Source Path (Local or UNC)** setting, click the ellipsis button (**...**). The **Browse for Directory** dialog box opens. By default, the files are stored in IISROOTFOLDER.
9. Enter the same target directory as the one that contains the new file that you added in the **Files and Folders** view.
10. Click **OK**.

At installation, files are copied to the target directory folder. In addition, the virtual directory is configured for that folder on the target system if IIS is present.

Removing Applications and Virtual Directories from the Internet Information Services View

Task ***To remove an application or a virtual directory from your installation:***

1. In the View List under **Configure the Target System**, click **Internet Information Services**.
2. In the **Web Sites** explorer, right-click the virtual directory and click **Delete**.

Feature Associations for IIS Support

InstallShield lets you associate a Web site with a feature in your project. If that feature is installed, the Web site and all of its applications and virtual directories are installed. If the selected feature is not installed, the Web site and its applications and virtual directories are not installed.

Task ***To associate a Web site with a feature in your project:***

1. In the View List under **Configure the Target System**, click **Internet Information Services**.
2. In the explorer, click the Web site that you want to associate with a feature.
3. In the **Feature** setting, select the name of the existing feature that should contain the selected IIS data.

Tip ▪ *If you delete a feature in your project, the Web site, applications, and virtual directories that are associated with the feature are simultaneously removed from your project.*

Uninstalling Web Sites, Applications, and Virtual Directories

A Web site that is created by an installation is never removed unless both of the following conditions are true:

- The Web site no longer contains applications or virtual directories.
- The value for the **Delete on Uninstall** setting for the Web site in the Internet Information Services view is Yes.

If a feature is uninstalled, its Web site, its IIS applications, and its virtual directories are uninstalled.

Setting the ASP.NET Version for a Web Site or Application

InstallShield lets you set the ASP.NET version for a Web site or application in your installation. If you specify the ASP.NET version, after the Web site or application is created, the installation runs the ASP.NET IIS Registration tool (`Aspnet_regiis.exe`) to map the Web site or application to the version that you specify.

If you specify the ASP.NET version for a Web site, IIS uses that value for the Web site that is created at run time, as well as for any of its applications.

Important ▪ *Microsoft does not recommend using the* `Aspnet_regiis.exe` *tool on Windows Vista, Windows Server 2008, and later systems because it has limited capabilities. As a result, you may need to manually define application mappings in the Internet Information Services view. To learn more, see Defining Application Mappings for a Web Site, Application or Virtual Directory.*

ASP.NET 3.0 does not include the `Aspnet_regiis.exe` *tool. Therefore, you cannot set the ASP.NET version to version 3 of ASP.NET.*

Task ***To specify the ASP.NET version for a Web site or application in your project:***

1. In the View List under **Configure the Target System**, click **Internet Information Services**.
2. In the **Web Sites** explorer, select the Web site or application whose ASP.NET version you want to specify. The settings for the Web site or application are displayed on the right.
3. For the **ASP.NET Version** setting, enter the complete version number of the .NET Framework that is required by your application, or select it from the list.

   For example, to specify version 2 of ASP.NET, type **`2.0.50727`**. To specify version 1.1 of ASP.NET, type **`1.1.4322`**.

Defining Application Mappings for a Web Site, Application or Virtual Directory

InstallShield lets you define mappings between file name extensions and the applications that process those files.

Task ***To add, edit, or remove an application mapping:***

1. In the View List under **Configure the Target System**, click **Internet Information Services**.
2. In the **Web Sites** explorer, select the Web site, application, or virtual directory that you want to configure.
3. In the **Application Mappings** setting, click the ellipsis button (**...**). The **Application Mappings** dialog box opens.
4. Do one of the following:
   a. To add a new mapping, click the **Add** button. The **Application Extension Mapping** dialog box opens. For more information, see Application Extension Mapping Dialog Box.
   b. To change an existing mapping, select the one that you want to edit, and then click the **Edit** button.
   c. To delete an existing mapping, select it and then click the **Delete** button.

Specifying Timeout Parameters for a Web Site or Virtual Directory

InstallShield lets you specify timeout parameters for a Web site, application, or virtual directory.

Task ***To specify timeout parameters:***

1. In the View List under **Configure the Target System**, click **Internet Information Services**.
2. In the **Web Sites** explorer, select the Web site, application, or virtual directory that you want to configure.
3. In the **Application settings** area, click the **Configuration** button. The **Application Mappings** dialog box opens.
4. In the **Session Timeout (minutes)** and the **ASP Script Timeout (seconds)** settings, specify the appropriate timeout values.

Configuring Additional IIS Virtual Directory Settings

When you select a virtual directory in the Web Sites explorer in the Internet Information Services view, the tabs that are displayed expose the most common settings for IIS virtual directories. You can configure other IIS virtual directory settings that are not exposed on those tabs.

Task ***To configure a setting that is not exposed in the Internet Information Services view:***

1. Create a VBScript deferred custom action.
2. In the View List under **Define Setup Requirements and Actions**, click **Custom Actions**.
3. Add the custom action to the **After Setup Complete Success** dialog item in the **Custom Actions during Installation** explorer.
4. Use the ADSI object model to configure the settings.

Refer to the MSDN documentation on how to use the ADSI object model for configuring IIS virtual directory settings.

Configuring Custom Error Messages for a Web Site, Application, or Virtual Directory

When an end user tries to connect to a Web site and a hypertext transfer protocol (HTTP) error occurs, the end user's browser displays a default message describing the error. You can have your installation configure IIS so that it uses custom error messages instead of the default error messages by mapping the HTTP error codes to a file or a URL.

Task ***To configure custom error messages for a Web site, application, or virtual directory:***

1. Create a file that contains your custom error message and add it to your installation.
2. In the View List under **Configure the Target System**, click **Internet Information Services**.
3. Select the Web site, application, or virtual directory for which you want to customize HTTP error messages. The settings for the Web site, application, or virtual directory are displayed on the right.
4. In the **Custom Errors** setting, click the ellipsis button (**...**). The **Custom Errors** dialog box opens.

5. Select the HTTP error code that you want to change and then click the **Edit Properties** button. The **Error Mapping Properties** dialog box opens.
6. To map the error code to a file:

   a. In the **Message Type** list, select **File**.

   b. In the **File** box, type the path and file name that points to the custom error message in your installation, or click the browse button to locate the file.

   To map the error code to a URL:

   a. In the **Message Type** list, select URL.

   b. In the **URL** box, type the URL that points to your custom error message.

Deploying Web Services on a Target Machine

Deploying a Web service onto a target system requires copying the Web service–specific files to a particular location and assigning a virtual directory name for that folder so that the Web service can be accessed via HTTP.

Tip ▪ *To learn how to add a virtual directory to your project, see Creating a Web Site and Adding an Application or a Virtual Directory.*

Task ***To deploy Web services on a target machine:***

1. In the View List under **Specify Application Data**, click **Files**.
2. In the **Feature** list, select the feature with which you want the Web services associated.
3. In the **Destination computer's folders** pane, select the folder (target directory) for installing files on the target system. Add your files to this folder. You may want this folder to be a new folder that you create in the folder called [IISROOTFOLDER].
4. In the View List under **Configure the Target System**, click **Internet Information Services**.
5. In the **Web Sites** explorer, select the virtual directory that is associated with the Web service.
6. In the **Content Source Path (Local or UNC)** setting, click the ellipsis button (**...**). The **Browse for Directory** dialog box opens. Enter the same target directory as the one that contains the new files that you added in the **Files and Folders** view.

At installation, files are copied to the target directory folder. In addition, the virtual directory is configured for that folder on the target system if IIS is present.

Stopping IIS Functionality Through a Custom Action

Stopping IIS through a custom action lets your installation overwrite files that are locked by IIS. If you are using one of the recent versions of IIS, you can run the following VBScript to unload a Web site, allowing you to install your .dll files. This means that you do not have to restart IIS, which will drop all of the Web sites that are running.

For example:

```
Dim DirObjSet DirObj = GetObject("IIS://LocalHost/W3SVC/1/Root/")DirObj.AppUnloadset dirObj = nothing"
```

Enabling Forms Authentication on Web Applications

Project *▪ This information applies to the following project types:*

- *Basic MSI*
- *InstallScript MSI*

You can use the **Forms Authentication** setting, displayed under the **Authenticated Access** section of the **Internet Information Services** view for a Web site, to set forms authentication on web applications.

ASP.NET forms-based authentication works well for sites or applications on public Web servers that receive many requests. This authentication mode lets you manage client registration and authentication at the application level, instead of relying on the authentication mechanisms provided by the operating system.

Important *▪ Forms authentication sends the user name and password to the Web server as plain text. You should use Secure Sockets Layer (SSL) encryption for the Log On page and for all other pages in your application except the Home page.*

Task ***To enable forms authentication on a Web site:***

1. In the View List under **Configure the Target System**, click **Internet Information Services**.
2. In the **Web Sites** explorer, select a Web site.
3. Under **Security > Authenticated Access**, set the **Forms Authentication** setting to **Yes**.

Adding IISROOTFOLDER Support

IISROOTFOLDER is an InstallShield directory variable that is used to determine the root directory of the Web server on a target system. If you are using IIS functionality in your installation project and you have added a Web site, then IISROOTFOLDER is automatically added.

Note *▪ All of the files that you add to the IISROOTFOLDER directory in the Files view are installed to the Web server's root directory on the target machine. If IIS is not on the target machine, the files are copied to the root folder.*

IIS_WEBSITE_NAME Property

The `IIS_WEBSITE_NAME` property is obsolete. If this property exists from earlier project versions, the upgrader will handle it automatically. The upgrader will create a Web site and set the site number field to [IIS_WEBSITE_NAME]. For new Web sites, you can use any property or hard-coded number.

IIS_PORT_NUMBER Property

The `IIS_PORT_NUMBER` property is obsolete. If this property exists from earlier project versions, the upgrader will handle it automatically. The upgrader will create a Web site and set the port number field to [IIS_PORT_NUMBER]. When you create new Web sites, you can use any property or hard-coded number.

Defining the End-User Interface

This section of the help library covers several features of InstallShield that enable you to define different aspects of the end-user interface. Everything from dialogs and billboards to the text displayed during the installation is covered.

Working with Dialogs

Your installation's user interface is important in many ways, primarily because end-user input and settings are usually handled through the user interface. If your user interface is difficult to navigate and understand, users may experience problems installing your product. To ease the process of creating your installation and to improve the end-user experience, InstallShield provides several predefined dialogs.

Although you are limited to the provided dialogs, you can customize many of them to produce the look and functionality that you require. For example, you can add a custom image to the top of every dialog, thereby branding it with your company logo.

Adding a Dialog to an Installation

Task ***To add a dialog to your installation:***

1. In the View List under **Customize the Setup Appearance**, click **Dialogs**.
2. In the **Dialogs** explorer, select the check box for the dialog that you want to add.

Dialog Themes

Project ▪ *Dialog themes are available in Express projects.*

Dialog themes are predefined sets of images that give your end-user dialogs a unified and distinctive look.

By changing the theme option selected in the Global Dialog Theme setting in the Dialogs view, you can select one of the available themes for your project, and InstallShield applies that theme to all of the interior and exterior dialogs, as well as the `Setup.exe` initialization dialog, in your project.

Note ▪ *InstallShield does not currently let you create your own dialog themes. However, InstallShield includes two different themes. For more information, see Available Themes.*

Selecting or Changing a Dialog Theme

Project ▪ *Dialog themes are available in Express projects.*

You can use a dialog theme to change the look of the end-user dialogs in your installation. You can select one theme per project.

Task ***To change the dialog theme used for a project:***

1. In the View List under **Customize the Setup Appearance**, click **Dialogs**.
2. Click the **Dialogs** explorer. The global dialog settings are displayed in the right pane.
3. In the **Global Dialog Theme** setting, select the theme that you want to use.

InstallShield applies the selected theme to the dialogs in your project.

Available Themes

Project ▪ *The Premier and Professional editions of InstallShield include additional dialog themes that are not available in the Express edition.*

InstallShield includes two different themes.

- Classic Theme
- InstallShield Blue Theme

To view samples of both themes, see the End-User Dialogs section of the documentation.

Bitmap Images in Dialogs

Each end-user dialog contains one of several images, as described below.

Splash Bitmap

The splash bitmap image is displayed on the Splash Bitmap dialog. This image must be a .bmp or .jpg file, and it must be 465 pixels wide by 281 pixels high.

The following sample dialogs—one with the Classic theme and one with the InstallShield Blue theme—show the default splash bitmap file. You can replace the default image with your own image file.

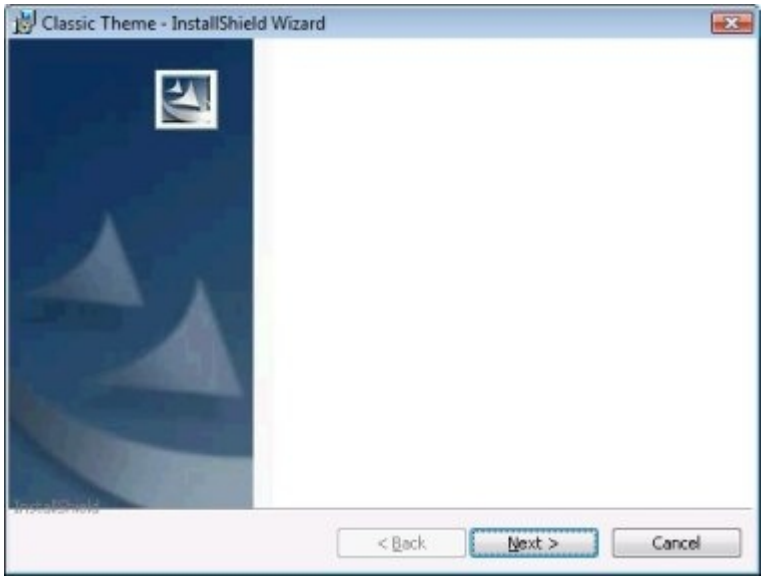


Figure 4-1: Splash Bitmap Dialog with Classic Theme

Figure 4-2: Splash Bitmap Dialog with InstallShield Blue Theme

Bitmap Image

The bitmap image is displayed in the background of the Install Welcome dialog and the Setup Complete Success dialog. This image must be a .bmp or .jpg file, and it must be 499 pixels wide by 312 pixels high.

You can change the bitmap image for each individual dialog, or you can specify a global dialog image that should be used in all of the dialogs that contain the bitmap image.

The following sample dialogs—one with the Classic theme and one with the InstallShield Blue theme—show the default bitmap image. The image includes the column on the left, as well as the white background behind the text.

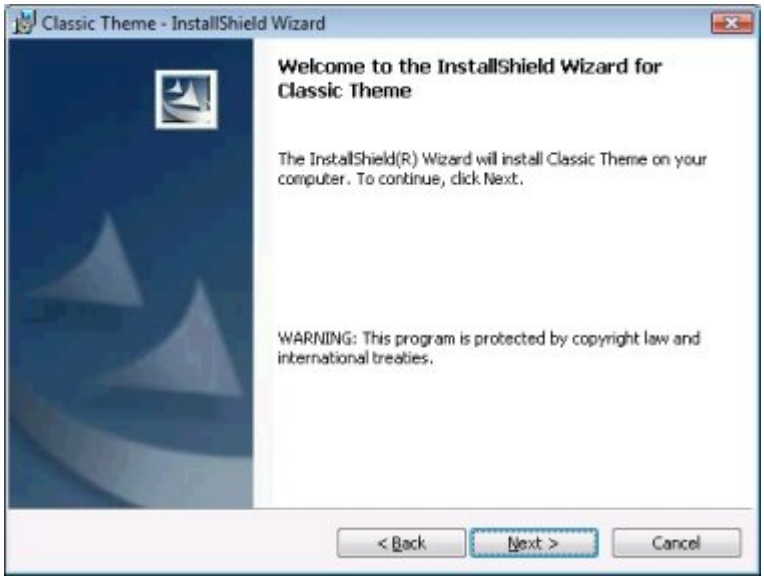


Figure 4-3: Install Welcome Dialog with Classic Theme

Figure 4-4: Install Welcome Dialog with InstallShield Blue Theme

Banner Bitmap

The banner bitmap image is displayed directly below the title bar of several dialogs: License Agreement, Readme, Customer Information, Destination Folder, Database Folder, Setup Type, Custom Setup, Ready to Install, and Setup Progress. This image must be a .bmp or .jpg file, and it must be 499 pixels wide by 58 pixels high.

You can change the banner bitmap for each individual dialog, or you can specify a global dialog banner that should be used in all of the dialogs that contain the banner bitmap.

The following sample dialogs—one with the Classic theme and one with the InstallShield Blue theme—show the banner bitmap. The image consists of the graphic at the far-right corner of the dialog title area.

Figure 4-5: License Agreement Dialog with Classic Theme

Figure 4-6: License Agreement Dialog with InstallShield Blue Theme

Changing the Splash Bitmap in the Splash Bitmap Dialog

Task ***To change the splash bitmap image on the Splash Bitmap dialog:***

1. In the View List under **Customize the Setup Appearance**, click **Dialogs**.
2. In the **Dialogs** explorer, click **Splash Bitmap**.

3. In the **Splash Bitmap** setting, type the path to the .bmp or .jpg file that you want to use as the dialog's image. Alternately, click the ellipsis button (**...**) to browse to the file. The image must be 465 pixels wide by 281 pixels high.

Changing the Bitmap Image for an End-User Dialog

Task ***To change the bitmap image for an end-user dialog:***

1. In the View List under **Customize the Setup Appearance**, click **Dialogs**.
2. In the **Dialogs** explorer, click the dialog whose bitmap image you want to change.
3. In the **Bitmap Image** setting, type the path to the .bmp or .jpg file that you want to use as the dialog's image. Alternately, click the ellipsis button (**...**) to browse to the file. The image must be 499 pixels wide by 312 pixels high.

Important ▪ *You can also configure the bitmap image for all end-user dialogs by configuring the Global Dialog Image setting, which is displayed when you click the Dialogs explorer in the Dialogs view. If you change the value of the Bitmap Image setting for a particular dialog and then you change the value of the Global Dialog Image setting, the value in the Bitmap Image setting is overwritten with the value in the Global Dialog Image setting.*

Changing the Banner Bitmap for an End-User Dialog

Task ***To change the banner bitmap for an end-user dialog:***

1. In the View List under **Customize the Setup Appearance**, click **Dialogs**.
2. In the **Dialogs** explorer, click the dialog whose banner bitmap you want to change.
3. In the **Banner Bitmap** setting, type the path to the .bmp or .jpg file that you want to use as the dialog's image. Alternately, click the ellipsis button (**...**) to browse to the file. The image must be 499 pixels wide by 58 pixels high.

Important ▪ *You can also configure the banner bitmap for all end-user dialogs by configuring the Global Dialog Banner setting, which is displayed when you click the Dialogs explorer in the Dialogs view. If you change the value of the Banner Bitmap setting for a particular dialog and then you change the value of the Global Dialog Banner setting, the value in the Banner Bitmap setting is overwritten with the value in the Global Dialog Banner setting.*

Changing the Global Dialog Image for End-User Dialogs

Task ***To change the bitmap image for all end-user dialogs that contain it:***

1. In the View List under **Customize the Setup Appearance**, click **Dialogs**.
2. Click the **Dialogs** explorer.
3. In the **Global Dialog Image** setting, type the path to the .bmp or .jpg file that you want to use as the bitmap image in all dialogs that contain it. Alternately, click the ellipsis button (**...**) to browse to the file. The image must be 499 pixels wide by 312 pixels high.

Tip ▪ *You can also set the bitmap image for individual dialogs.*

Caution ▪ *If you change the value of the Bitmap Image setting for a particular dialog and then you change the value of the Global Dialog Image setting, the value in the Bitmap Image setting is overwritten with the value in the Global Dialog Image setting.*

Changing the Global Dialog Banner for End-User Dialogs

Task ***To change the banner bitmap for all end-user dialogs that contain it:***

1. In the View List under **Customize the Setup Appearance**, click **Dialogs**.
2. Click the **Dialogs** explorer.
3. In the **Global Dialog Banner** setting, type the path to the .bmp or .jpg file that you want to use as the banner bitmap in all dialogs that contain it. Alternately, click the ellipsis button (**...**) to browse to the file. The image must be 499 pixels wide by 58 pixels high.

Tip ▪ *You can also set the banner bitmap for individual dialogs.*

Caution ▪ *If you change the value of the Banner Image setting for a particular dialog and then you change the value of the Global Dialog Banner setting, the value in the Banner Image setting is overwritten with the value in the Global Dialog Banner setting.*

Minimizing Reboots on Windows Vista and Later Systems

Windows Logo ▪ *Restarting the system after an installation is inconvenient for end users. One of the Windows logo program requirements is that all installations must contain an option that enables end users to automatically close applications and attempt to restart them after the installation is complete.*

To support this requirement, all Express projects include the MsiRMFilesInUse dialog by default. An installation displays the MsiRMFilesInUse dialog on a Windows Vista or later system if one or more files that need to be updated are currently in use during the installation. The dialog contains two options to allow end users to specify how to proceed:

- End users can choose to have the installation close the applications that are using those files and then attempt to restart the applications after the installation is complete.
- End users can avoid closing the applications. A reboot will be required at the end of the installation.

For the best end-user experience, your application should be instrumented to use the Restart Manager API; doing so allows the Restart Manager to effectively pause and resume your application exactly where the end user left it. For detailed information, see About Restart Manager and the other Restart Manager documentation on the MSDN Web site.

Custom Setup Dialog Options

The Custom Setup dialog has a sophisticated user interface that is tightly integrated with information about the target system, the features in your installation, and Windows Installer installation options. It provides the end user with the most control over the installation.

Many of the options and information it offers are determined by the feature properties that you set in your setup design, as described below.

Table 4-1 ▪ Custom Setup Dialog Options

Option	Description
Advertisement	Feature advertisement allows files to be installed on demand after the installation has initially run. In the Custom Setup dialog, when you click a feature, you can specify that you want it installed later by selecting the Will be installed when required option. However, that default option is present only if the setup author selects Yes for the feature's Advertised setting.
Hide a Feature from the End User	When you set a feature's Visible setting to No, the end user cannot see the feature or its subfeatures in the Custom Setup dialog and therefore cannot change any of its installation options.
Display All Subfeatures	The feature's Visible setting also governs whether its subfeatures are expanded when the dialog first opens.

Table 4-1 ▪ Custom Setup Dialog Options (cont.)

Option	Description
Display a Description for a Feature	The description that is shown at the bottom of the Custom Setup dialog when you select a feature is taken from its Description setting.
Change the Feature Order	You can change the order in which the features appear to the end user in the Custom Setup dialog. The order in which features are displayed is dictated by the order in the Features view. For more information, see Changing the Feature Order in the Custom Setup Dialog.
Require a Feature Be Installed	If you set a feature's Required setting to Yes, the end user will not have the **Will not be available** option, meaning that the feature must be installed.

Displaying a License Agreement During the Run Time

A license agreement dialog is included as part of the default user interface of installations. To display your license agreement text, associate a rich text file (.rtf) with this dialog.

Task ***To add your text to the license agreement dialog:***

1. In the View List under **Customize the Setup Appearance**, click **Dialogs**.
2. In the **Dialogs** explorer, click **License Agreement**.
3. Click the **License File** setting, and then click the ellipsis button (**...**) to browse to the .rtf file that you want to use.

Removing a Dialog from an Installation

Task ***To remove a dialog from your installation:***

1. In the View List under **Customize the Setup Appearance**, click **Dialogs**.
2. In the **Dialogs** explorer, clear the check box for the dialog that you want to delete.

Editing Run-Time Text and Message Strings

InstallShield provides you with complete control over all of the localizable strings displayed in your installation. You can customize everything from the text on the Next buttons to the descriptions of features.

The Text and Messages view in InstallShield lists all of the dialogs and messages that can be displayed during the installation process. If you click one of the dialogs in this view, you can see a sample screen shot of the dialog, as well as a string table that contains all of the text strings that belong to the dialog. Likewise, if you click a message in this view, you can see a sample message box and the string table with all of its associated strings. All run-time strings are accessible in this view.

Localization for international audiences is an important aspect of creating installations, and InstallShield enables you to export all of the run-time strings in your installation project to a text (.txt) file to ease translation. When you export the strings, they can be sent for translation in one common file type and can then be imported back into your installation project for a localized user interface.

Editing Run-Time Strings

Task ***To edit a run-time string:***

1. In the View List under **Customize the Setup Appearance**, click **Text and Messages**.
2. In the **Text and Messages** explorer, click the dialog or message that you want to edit.
3. Double-click the value of the string that you want to edit and type the new string.

Tip ▪ *You can use Windows Installer properties in your dialog text. To do this, surround the property with square brackets—*`[INSTALLDIR]`*, for example.*

Adding Comments to Text and Message Strings

You can add comments to a text or message string in your project. The end user does not see these comments. They exist only to assist in identifying strings.

Task ***To add comments to a string:***

1. In the View List under **Customize the Setup Appearance**, click **Text and Messages**.
2. In the **Text and Messages** explorer, click the dialog or message to which you want to add a comment, and then find the string that should have the comment.
3. Double-click the **Comment** value for that string and then type your comment. You may need to scroll to the right to see this value.

Changing the Font for Text and Message Strings

The default font and font size for all of the strings in your installation's dialogs and message boxes are set in the General Information view. InstallShield enables you to override the default font or font size for one or more specific strings.

Task ***To change the font and font size for a specific string:***

1. In the View List under **Customize the Setup Appearance**, click **Text and Messages**.
2. In the **Text and Messages** explorer, click the dialog or message that contains the string whose font you want to modify, and then find the appropriate string.

3. Double-click the **Font** value for the string whose font you want to modify.
4. Do one of the following:
   - In the **Font** list, select the appropriate font.
   - In the **Font** setting, click the ellipsis button (**...**). The **Font** dialog box opens. Select the font and font characteristics (such as size, style, and color) for the selected string, and then click **OK**.

Exporting Strings

To ease the task of translating all of the run-time strings in your installation project, InstallShield enables you to export the strings to a text (.txt) file. You can provide that .txt file to a translator who can update the file with translated text. Then you can import the .txt file back into your installation project for a localized user interface.

Task ***To export all of the run-time strings in your project:***

1. On the **Project** menu, click **Export String Entries**. The **Export String Table Wizard** opens.
2. Complete the panels in the wizard to export the strings to a .txt file.

Importing String Tables

Once the .txt file containing your installation's strings has been translated, you can import the .txt file into your project for a localized user interface.

Task ***To import the translated run-time strings in your project:***

1. On the **Project** menu, click **Import String Entries**. The **Import String Table Wizard** opens.
2. Complete the panels in the wizard to import the strings.

Displaying Billboards

You can add billboards to your projects to display information to end users during the installation process. The billboards can be used to communicate, advertise, educate, and entertain end users. For example, billboards can present overviews on new features of the product being installed or other products from your company. Each billboard is a file that you or your company's graphics department creates for complete control over the look and feel of the file transfer.

Billboard File Types

InstallShield supports different types of files for billboards:

- Adobe Flash application file (.swf)
- Images (.bmp, .gif, .jpg, and .jpeg)

If a target system does not have the Adobe Flash Player, which is used to display Flash application files, the installation can detect that and display image billboards in place of the Flash billboard. Therefore, if you include a Flash billboard in your project, it is recommended that you also include one or more image billboards in your project.

Note ▪ *You can add more than one image billboard to a project, but only one Adobe Flash application file billboard.*

Types of Billboards

InstallShield offers support for three different billboard styles. For example, with one style, the installation displays a full-screen background, with billboards in the foreground, and a small progress box in the lower-right corner of the screen. With another style, the installation displays a standard-size dialog that shows the billboards. The bottom of this dialog shows the progress bar.

Following are descriptions and sample screen shots of each type of billboard.

Fullscreen with Small Progress (Displayed in Lower Right)

For the **Fullscreen with Small progress (displayed in lower right)** type of billboard, when the installation displays the standard end-user dialogs, it also displays a full-screen background. During file transfer, the installation shows full-screen backgrounds, with billboards in the foreground, and a small progress box in the lower-right corner of the screen.

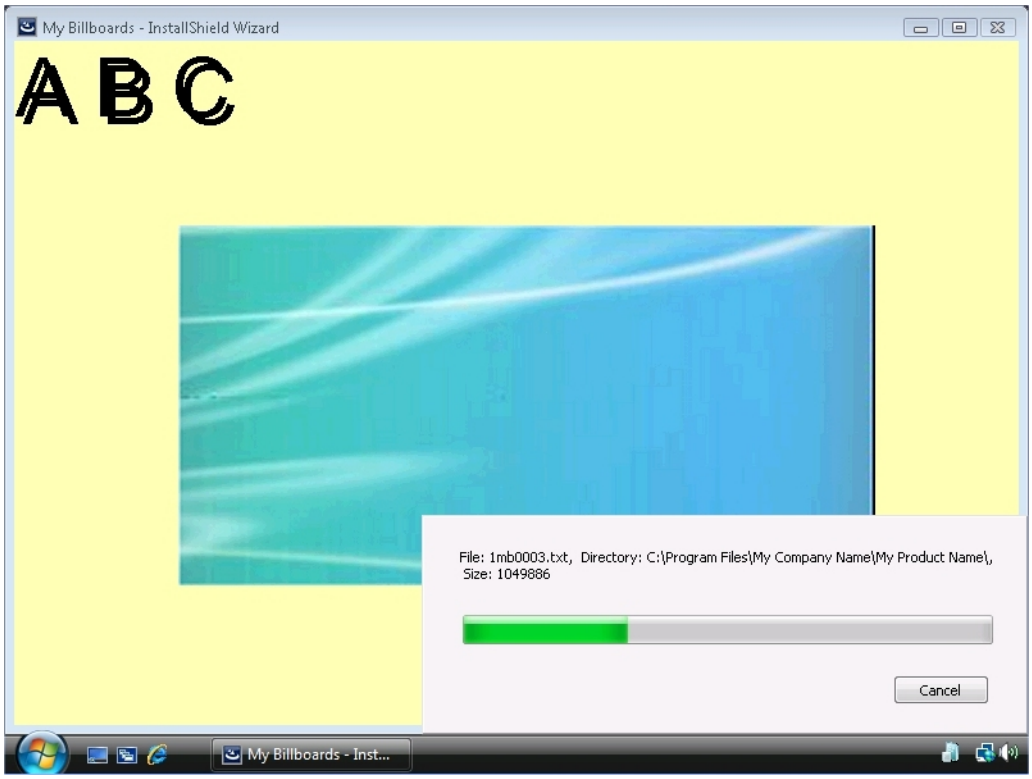


Figure 4-7: Fullscreen Billboard with Small Progress (Displayed in Lower Right)

In the sample screen shot, the billboard is the blue-green rectangle image in the center. Some of the configurable billboard settings were set as follows:

- Origin—Centered
- Title—A B C
- Font—48 pt. Arial
- Background Color—Yellow

Windowed with Standard Progress

For the **Windowed with Standard progress** type of billboard, during file transfer, the installation displays a standard-size dialog that shows the billboards. The bottom of this dialog shows the progress bar. The installation does not display a background for this style.

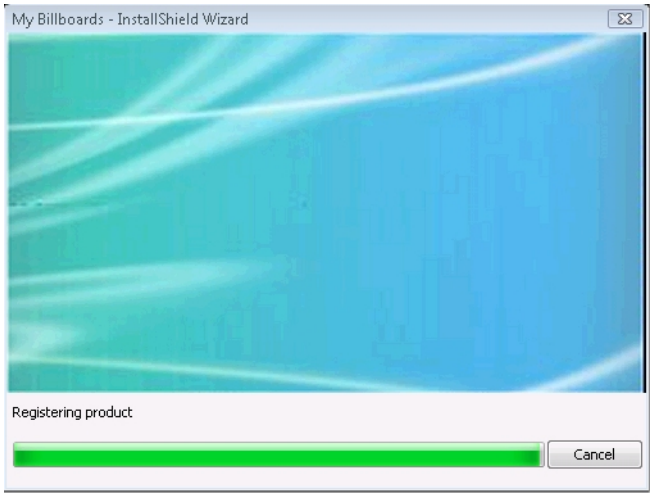


Figure 4-8: Windowed with Standard Progress

In the sample screen shot, the billboard is the blue-green rectangle image. Its size is 544 pixels wide by 281 pixels high.

Windowed with Small (Displayed in Lower Right, No Billboards)

For the **Windowed with Small (displayed in lower right, no billboards)** type of billboard, the installation displays a small progress box in the lower-right corner of the screen during file transfer. It does not display any billboards or a background.

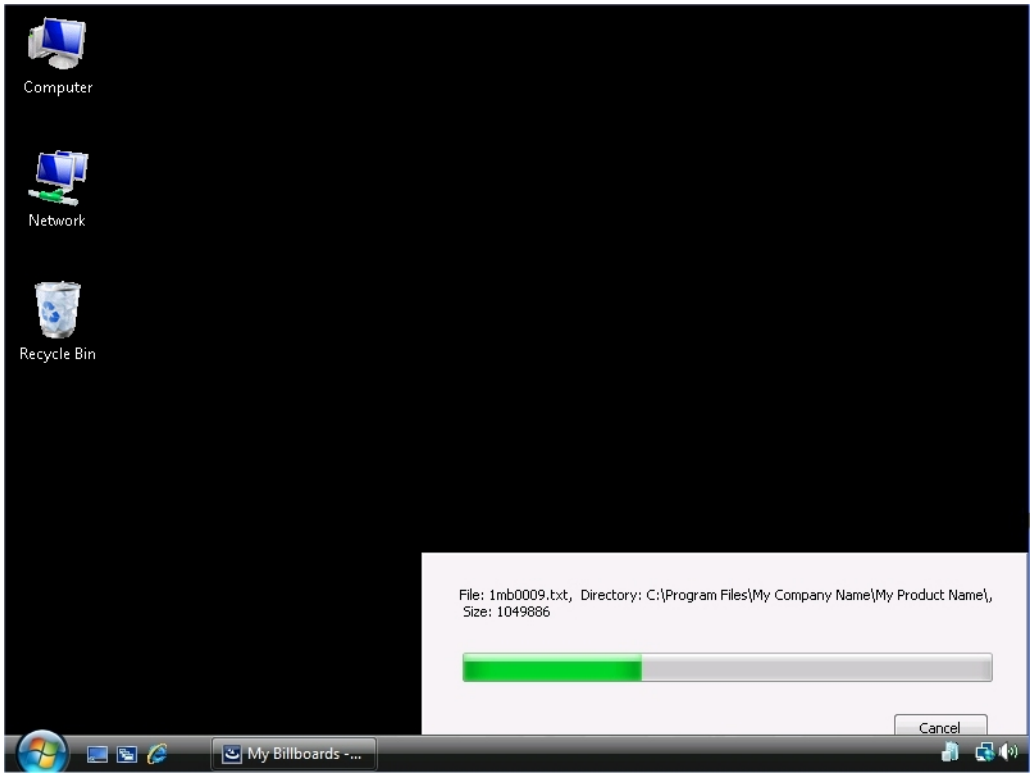


Figure 4-9: Windowed with Small (Displayed in Lower Right, No Billboards)

As shown in the sample screen shot, the progress bar is shown, but no billboard is displayed. The black background is the end user's desktop.

Specifying Which Type of Billboard to Use in an Installation

InstallShield offers support for different billboard styles.

Task ***To specify which type of billboard you want to use in your installation:***

1. In the View List under **Customize the Setup Appearance**, click **Billboards**.
2. In the center pane, click the **Billboards** explorer. InstallShield displays the **Billboard Type** setting in the right pane.
3. In the **Billboard Type** setting, select the appropriate style of billboard.

To see samples of each type of billboard, see Types of Billboards.

Adding an Adobe Flash Application File Billboard

InstallShield lets you display a Flash application file billboard during the file transfer process. Flash application files can consist of videos, movies, sounds, interactive interfaces, games, text, and more—anything that is supported by the .swf type of file. It is recommended that files such as Flash video files (.flv) and MP3 audio files

be embedded in the .swf file so that they are available locally on the target system during file transfer. Although .swf files can reference external files that you can post on a Web site, this external implementation would require that end users have an Internet connection.

Task ***To add an Adobe Flash Application File billboard to your installation:***

1. In the View List under **Customize the Setup Appearance**, click **Billboards**.
2. In the **Billboards** explorer, right-click **Adobe Flash Application File (.swf)** and then click **New Billboard**. InstallShield adds a new billboard item with the name **NewBillboard1**.
3. Type a name for the billboard item. This name is used to help you identify the item while you are creating your installation; the name is not displayed during the installation.
4. In the right pane, configure the settings for the billboard.

Note ▪ *If the version of Flash or other tool that you use to create your .swf file is newer than the version of the Flash Player that is installed on a target system, it is possible that some of the Flash features may not work as expected on that target system.*

Adding Image Billboards

You can choose to display only one image billboard during the file transfer process, or you can include a series of image billboards, each one designed to be displayed for a specific amount of time. InstallShield includes support for .bmp, .gif, .jpg, and .jpeg image files.

Note ▪ *Animated .gif files are not supported. If you want to use animation in a billboard, consider using an Adobe Flash application file billboard.*

Task ***To add an image billboard to your installation:***

1. In the View List under **Customize the Setup Appearance**, click **Billboards**.
2. In the **Billboards** explorer, right-click **Images** and then click **New Billboard**. InstallShield adds a new billboard item with the name **NewBillboard1**.
3. Type a name for the billboard item. This name is used to help you identify the item while you are creating your installation; the name is not displayed during the installation.
4. In the right pane, configure the settings for the billboard.

Configuring Billboard Settings

When you add an Adobe Flash application file billboard or an image billboard to your project, you need to configure its settings.

Task ***To configure a billboard's settings:***

1. In the View List under **Customize the Setup Appearance**, click **Billboards**.
2. In the **Billboards** explorer in the center pane, click the billboard that you want to configure. InstallShield displays the billboard settings in the right pane.
3. Configure the settings as required.

For information about each of the billboard settings, see Settings for Adobe Flash Application File Billboards and Image Billboards.

Previewing Billboards Without Building and Launching a Release

InstallShield lets you preview a billboard to see how it would be displayed at run time, without requiring you to build and run a release.

Previewing a billboard lets you see how your billboard will look with the background color, position, and related settings that are currently configured for your billboard.

Task ***To preview a billboard:***

1. In the View List under **Customize the Setup Appearance**, click **Billboards**.
2. In the **Billboards** explorer in the center pane, right-click the billboard that you want to preview, and then click **Preview Billboard**.

InstallShield displays a preview of the billboard as it would be displayed at run time.

To stop previewing a billboard, click the Cancel button in the preview window.

Tip ▪ *Previewing a billboard is especially helpful if you want to see how your Flash or image billboard will look with different selected billboard types. You can preview a billboard, change the billboard type, and then preview a billboard again.*

Setting the Billboard Order

Image billboards are displayed in the order in which they are listed in the Billboards view, from top to bottom.

Task ***To change the order in which image billboards are displayed at run time:***

1. In the View List under **Customize the Setup Appearance**, click **Billboards**.
2. In the **Billboards** explorer, right-click one of the billboard items that you want to move, and then click either **Move Up** or **Move Down**.

Repeat the last step until all of the billboards are correctly sorted.

Run-Time Behavior of an Installation that Includes Billboards

Important ▪ *If your installation includes billboards, your installation must include a* `Setup.exe` *setup launcher. The setup launcher is required because it displays the billboards at run time. The Setup.exe tab for a release in the Releases view is where you specify information such as whether you want to use a setup launcher. To learn more, see Setup.exe Tab.*

If your installation includes a Flash billboard and one or more image billboards, only one billboard type is displayed at run time during the file transfer process: the Flash billboard or the image billboards.

- If the Flash Player is present on the target system, the installation displays the Flash billboard.
- If the Flash Player is not present, the installation displays the image billboards.

The run-time behavior is slightly different, depending on whether the installation is displaying a Flash billboard or image billboards:

- **If the installation is displaying a Flash billboard**—When the file transfer is complete, the installation continues showing the Flash billboard until its duration has elapsed. Once the duration has elapsed, the installation stops displaying the billboard and shows the appropriate Setup Complete dialog.

  If the file transfer takes more time than you have allocated for the duration of the Flash billboard, the installation continues displaying the Flash billboard until file transfer ends.

- **If the installation is displaying image billboards**—When the file transfer is complete, the installation stops displaying the image billboards, even if other billboards are scheduled or the current billboard's duration has not elapsed. The installation then shows the appropriate Setup Complete dialog.

  If the file transfer takes more time than you have allocated for the billboards, the installation continues displaying the billboards until file transfer ends. If No is selected for the Loop Billboard setting in the Billboards view and the installation reaches the last billboard before the file transfer ends, the installation continues displaying that last image billboard until file transfer ends. Then the installation shows the appropriate Setup Complete dialog. If Yes is selected for this setting and the installation reaches the last billboard before the file transfer ends, the installation restarts the display of billboards from the beginning. The loop continues, if necessary, until the file transfer ends, and the Setup Complete dialog is displayed.

Note ▪ *If the version of Flash or other tool that you use to create your .swf file is newer than the version of the Flash Player that is installed on a target system, it is possible that some of the Flash features may not work as expected on that target system.*

Removing a Billboard

Task ***To remove a billboard from your installation:***

1. In the View List under **Customize the Setup Appearance**, click **Billboards**.
2. In the **Billboards** explorer, right-click the billboard that you would like to remove, and then click **Delete**.

Preparing Installations for Maintenance and Uninstallation

InstallShield lets your users rerun your installation to change program features or reinstall or remove your application. When users select your application in Add or Remove Programs in the Control Panel, the installation displays dialog boxes that let users do any of the following:

1. Install individual features that were not previously installed, and uninstall individual features.
2. Reinstall your application with the settings selected during the first installation.
3. Uninstall your application.

To modify, repair, or uninstall an application, the operating system must have some indication that the application is present. To allow this, an installation registers an application with the operating system so that it can be easily maintained or uninstalled. You enter the necessary information in the General Information view. For more information, see Specifying Installation Information.

In InstallShield installations, maintenance (that is, modification and repair) is handled automatically. In addition, uninstallation is handled automatically—with the sole exception of custom actions, which must either undo their own effect during uninstallation or have their effect undone by another custom action that runs only during uninstallation.

Removing Registry Data Created by Your Product

By default, your product's uninstaller removes only data that was created by your installation program.

You can use a special uninstallation flag in the Registry view. The uninstallation flag controls the registry data to be removed during uninstallation. In particular, using the Uninstall entire key flag on a registry key causes the key and all its values and subkeys to be removed during uninstallation. For more information, see Registry Flags.

Building, Testing, and Distributing Installations

Once you have configured the features, files, shortcuts, registry entries, end-user dialogs, and other elements of your installation project, you are ready to create and build a release for your installation. Building a release packages the content of your installation, creating a disk image that you can copy to your distribution media and distribute or deploy as needed.

Testing is an essential part of creating a reliable installation. InstallShield enables you to selectively test run just the end-user interface portion of a release. You can also run your installation by simply clicking a button in InstallShield; when you run an installation by using this method, your installation executes exactly as it would on an end user's machine. All files are transferred, shortcuts and registry entries are made, and the user interface is displayed.

The last step in creating your installation is to distribute it to a specified location. This location may be on a network drive, floppy disk, or different location on a local drive. When you distribute your installation, the disk image created when you built your installation is copied to the location that you specify.

Configuring and Building Releases

Once you have designed your project in InstallShield, you are ready to create a release that you can configure and build to distribute to end users. The release is built according to the options that you set for it in the Releases view.

Building a Release

Tip ▪ *The following instructions apply to releases that are built within InstallShield (without integration with Visual Studio). For information on building an InstallShield release from within Visual Studio, see Building Releases in Microsoft Visual Studio.*

Task ***To build a release:***

1. In the View List under **Prepare for Release**, click **Releases**.
2. In the **Builds** explorer, click the type of media that you want to build.
3. Edit the build settings.
4. Right-click the selected media type and then click **Build**.

You can also build a release at the command line using `IsCmdBld.exe`.

InstallShield places the built installation package into a release location based on your project location. If the build failed and generated an error, the previous installation package, if any, is restored to the `\DiskImages\Disk1` folder.

Tip ▪ *Ensure that Windows Explorer is not pointing to the Disk1 folder or a subfolder when you build your release. If it is pointing to the Disk1 folder, the build process will continue indefinitely. If Explorer is accessing a subfolder, you will generate an error.*

If you are creating non-compressed builds, additional folders named after specific operating systems may be included with the installation. This handles cases when files destined for a specific operating system have the same name as those targeted for another operating system.

Creating a Setup Launcher

InstallShield lets you specify whether you want your installation to include a `Setup.exe` setup launcher. A `Setup.exe` setup launcher is required in the following cases:

- You want to automatically update or install the Windows Installer engine on a target system, when necessary.
- Your project includes InstallShield prerequisites.
- Your project includes the .NET Framework.
- Your project includes billboards.

The `Setup.exe` setup launcher is a bootstrap application that manages the aforementioned scenarios.

The Setup.exe tab for a release in the Releases view is where you specify information such as whether you want to use a `Setup.exe` launcher. To learn more, see Setup.exe Tab.

Windows Installer and Setup.exe

If it is possible that Windows Installer is not present on a target system, or if your installation depends on certain functionality that is available in only a certain version of Windows Installer, InstallShield gives you the option of including with your installation a redistributable that installs Windows Installer. If you select this option, InstallShield creates a `Setup.exe` launcher that checks for the presence of Windows Installer on the target system. If Windows Installer is not installed, or a more recent version needs to be installed, `Setup.exe` launches the Windows Installer installation and then launches your installation package.

For more information, see Adding Windows Installer Redistributables to Projects.

InstallShield Prerequisites and Setup.exe

Projects that include InstallShield prerequisites require `Setup.exe` because `Setup.exe` checks to see if the target system meets the InstallShield prerequisite conditions. If the conditions are met, `Setup.exe` installs the InstallShield prerequisites. For more details, see Working with InstallShield Prerequisites that Are Included in Projects.

.NET Framework and Setup.exe

Projects that include the .NET Framework require `Setup.exe` because `Setup.exe` checks the target system for the presence of the .NET Framework; if the appropriate version of the .NET Framework is not present, `Setup.exe` installs it.

To learn more about including the .NET Framework, see Adding .NET Framework Redistributables to Projects.

Billboards and Setup.exe

Projects that include billboards require `Setup.exe` because `Setup.exe` displays the billboards at run time.

For more information about billboards, see Displaying Billboards.

Customizing File Properties for the Setup Launcher

InstallShield lets you use custom information for the version resources of the `Setup.exe` setup launcher. The information is displayed on the Properties dialog box for the setup launcher; this Properties dialog box opens when end users right-click the `Setup.exe` file and then click Properties.

Settings in InstallShield that Let You Configure Setup.exe Properties

The following table lists various properties that Windows includes on the Properties dialog box, along with the corresponding settings in InstallShield that you can use to configure them.

***Note** ▪ The Properties dialog box is different on different versions of Windows. For example, on Windows 7 systems, the version resource information is displayed on the Details tab of the Properties dialog box. However, on Windows XP systems, the version resource information is displayed on the Version tab of that dialog box.*

Also note that some versions of Windows do not show some settings on the Properties dialog box.

Table 4-1 ▪ Source of Information for Setup.exe Properties

Setup.exe Property	InstallShield Setting that Configures the Setup.exe Property
Company Name	Publisher setting in the General Information view
Product Name	Product Name setting in the General Information view
Product Version	Product Version setting in the General Information view.
File Version	File Version setting for the product configuration in the Releases view. If that setting is blank, InstallShield uses the Product Version setting in the General Information view. ***Note** ▪ The File Version always contains four fields. If you specify fewer than four fields for your File Version, the remaining fields are filled with zeros. For example, if you specify a File Version of 1.1, the File Version that is used in the version resources of Setup.exe is 1.1.0.0.*

Table 4-1 ▪ Source of Information for Setup.exe Properties (cont.)

Setup.exe Property	InstallShield Setting that Configures the Setup.exe Property
Copyright	InstallShield uses either a custom value that you provide, or the default InstallShield copyright notice. If you want InstallShield to use your own custom value for the copyright field: 1. In the **Releases** view, select the release that you want to configure. 2. On the **Setup.exe** tab, in the **Use Custom Version Properties** setting, select **Yes**. 3. In the **Launcher Copyright** setting, enter the text that you want to use for the copyright field in the properties of your `Setup.exe` file. If you select No for the Use Custom Version Properties setting or you leave the Launcher Copyright setting blank, InstallShield uses the default InstallShield copyright notice.
File Description	InstallShield uses either a custom value that you provide, or the default InstallShield `Setup.exe` description. If you want InstallShield to use your own custom value for the description field: 1. In the **Releases** view, select the release that you want to configure. 2. On the **Setup.exe** tab, in the **Use Custom Version Properties** setting, select **Yes**. 3. In the **File Description** setting, enter the text that you want to use for the description field in the properties of your `Setup.exe` file. If you select No for the Use Custom Version Properties setting, InstallShield uses the default InstallShield `Setup.exe` description. If you select Yes but leave the File Description setting blank, InstallShield uses the Summary Information Stream Comments setting in the General Information view. If that setting is blank, InstallShield uses the default InstallShield `Setup.exe` description.
Language	For this field, InstallShield uses the language that is used in your project.

Sample Setup.exe Properties Dialog Boxes

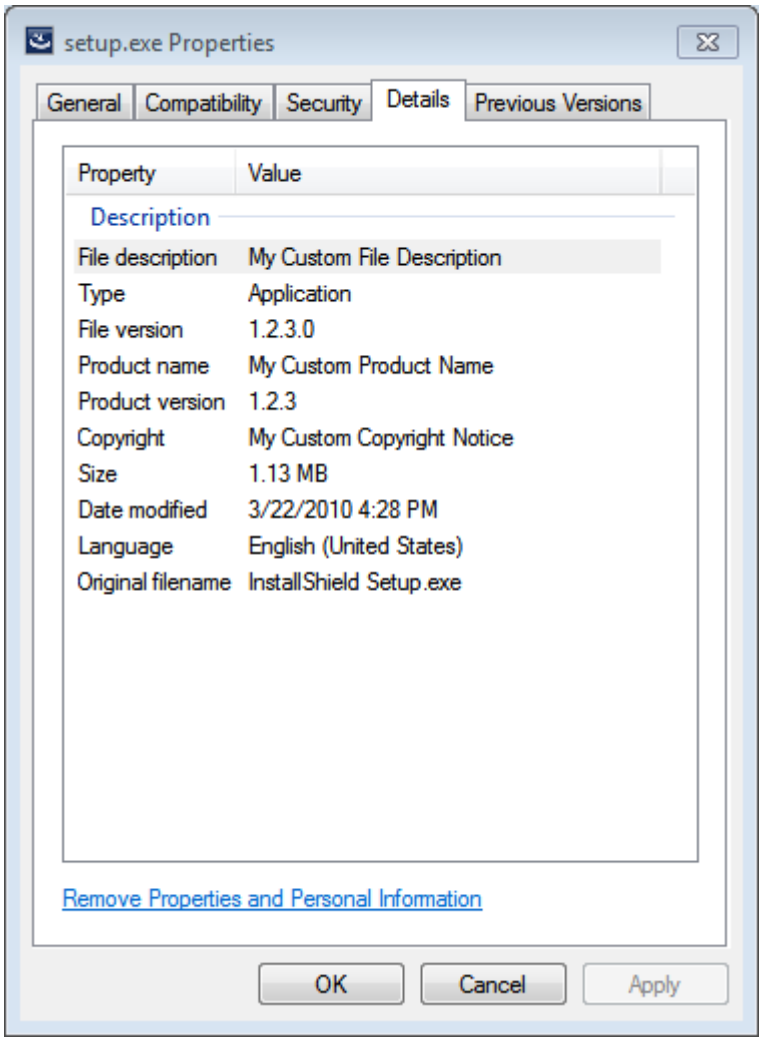


Figure 4-1: Sample Properties for Setup.exe on a Windows 7 Machine

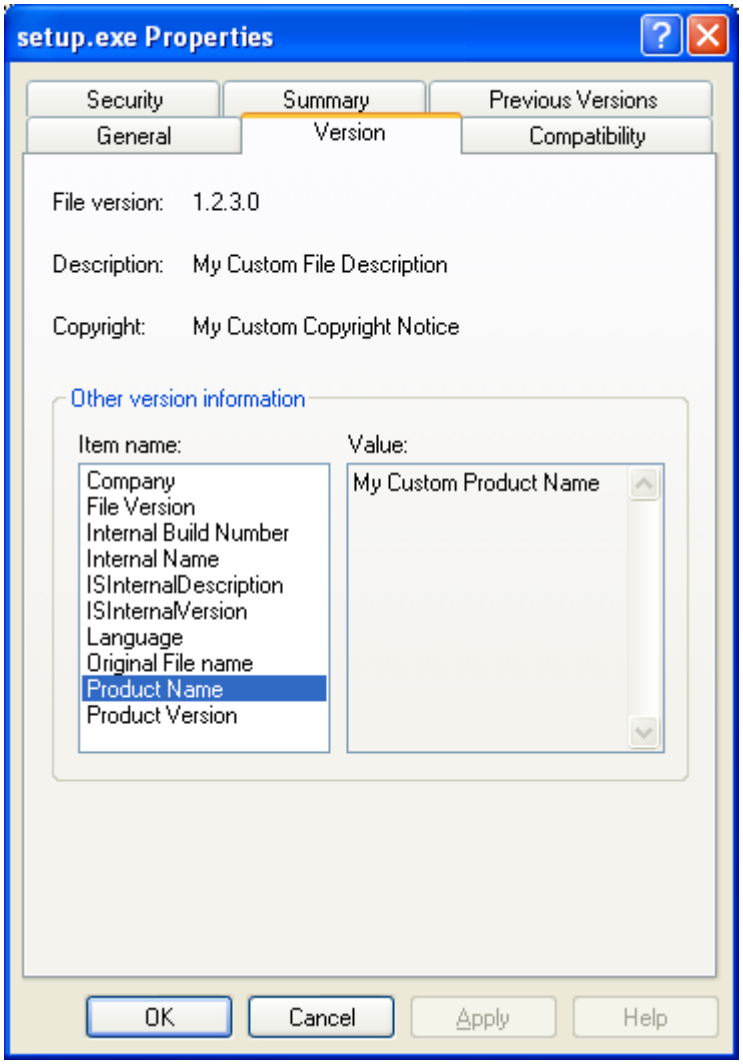


Figure 4-2: Sample Properties for Setup.exe on a Windows XP Machine

Building 64-Bit Setup Launcher

InstallShield lets you specify whether you want your installation to include a 64-bit **Setup.exe** setup launcher. A 64-Bit **Setup.exe** setup launcher is required only if the installation is targeting 64-bit system. By default, InstallShield builds the 32-bit setup launcher, which can execute in a 32-bit system as well in WoW64 on 64-bit system. Some 64-bit target systems-such as Windows Server Core systems-may not have 32-bit Windows-on-Windows (WoW64) support. These 64-bit target systems cannot run 32-bit setup launcher.

The **General** tab in the **Releases** view is where you specify information to build 64-Bit Setup launcher.

Building 64-bit Setup launcher does not build the corresponding 64-bit MSI package by default. In order to build the 64-bit MSI package, need to configure the Template Summary with the appropriate 64-bit value (x64 or Intel64). The Express edition of InstallShield configures the Template Summary property automatically. Pure 64-bit MSI package cannot copy 32-bit product files or execute 32-bit custom action during installation, because 32-bit binaries cannot be loaded on 64-bit target systems without WOW64 support.

For more information on Building 64-bit MSI package, see Challenges of Supporting Both 32-Bit and 64-Bit Target Operating Systems.

Building 32-bit MSI Package with 64-bit Setup Launcher

InstallShield throws the following warning message if the MSI package is configured to build 32-bit package with the 64-bit setup launcher option:

```
ISEXP: warning -7372: Setup Launcher is configured to build 64-Bit, but Template Summary is not configured to build 64-Bit MSI package.
```

This warning is received because the 64-bit setup launcher fails to execute in a pure 32-bit system, and the 32-bit MSI package fail to install in pure 64-bit system. This failure can be fixed by not opting the 64-bit setup launcher if the installer needs to target pure 32-bit system and WoW64 on 64-bit system. Alternatively, the MSI package can be configured to generate 64-bit MSI package to target pure 64-bit system.

InstallShield Prerequisites and 64-bit Setup Launcher

In general, projects that include InstallShield prerequisites require **Setup.exe** because **Setup.exe** checks to see if the target system meets the InstallShield prerequisite conditions. If the conditions are met, **Setup.exe** installs the InstallShield prerequisites.

If you configure to build the projects that include InstallShield prerequisites with 64-bit setup launcher, ensure the prerequisite installer is a 64-bit, which can install in pure x64 system and configure the prerequisite conditions to check the 64-bit location either in the registry or folder location.

In some cases, if you want to build the 32-bit prerequisite installer with 64-bit setup launcher, configure the conditions to check the 32-bit location. Most of the registry conditions in the prerequisites are configured to check the default location, which is 32-bit location for the 32-bit set up launcher and 64-bit location for the 64-bit setup launcher. But, the 32bit prerequisite installer creates the registry key under the 32-bit registry hives. To rectify this, you need to modify the registry location to check the 32-bit location explicitly.

Canceling Builds

Task ***To cancel a build after it has started:***

Click the **Stop Build** button on the toolbar.

Changing the Product Version During a Build

InstallShield offers different ways to change the product version for your installation at build time.

- If you are using `ISCmdBld.exe` to build your installation from the command line, use the `-y` command-line parameter to specify the product version. To learn more, see IsCmdBld.exe.

- If you are using MSBuild or Team Foundation Server to build your installation, use the ProductVersion parameter on the InstallShield task. For more information, see Microsoft Build Engine (MSBuild).

Build Release Location

The disk image folders for your installation are built in the release location. All additional files and folders necessary for storing uncompressed application files are placed in subfolders of the disk image folders. The release location is a subfolder of your project's location.

The release is built in the following folder:

```
<project location>\<project name>\Express\<release type>\DiskImages\Disk1
```

The default project location is:

```
C:\InstallShield 2022 Projects
```

For example, if the current project name is CoolProject, and the media type for the release is a CD-ROM, you can find the Disk1 folder under the following location:

```
C:\InstallShield 2022 Projects\CoolProject\Express\Cd_rom\DiskImages
```

If you are creating non-compressed builds, additional folders named after specific operating systems may be included with the installation. This handles cases when files destined for a specific operating system have the same name as those targeted for another operating system.

Build Logs and Reports

Each time you build a release, a log and build report are generated. The log contains the same information displayed in the Output window during the build process. The build report contains a concise summary of your build, as well as a listing of all features, setup types, merge modules, dynamic links, and files included in your build. Since both the log and report are generated and timestamped each time you build a release, you can use these for manual verification of the contents of setups, as well as for your records.

Quick Builds

When you are testing your installation, you might not want to continually build all of your installation files if no files have been changed. Once you have performed an initial complete build of your installation, you can use the Quick Build option in InstallShield. The Quick Build option enables you to rebuild only the .msi file part of an installation, shortening the build process. This is especially useful when you are testing or making changes to a string or dialog.

Note ▪ *The SingleImage media type does not support the Quick Build option because the .msi file is included in the* `Setup.exe` *file. In addition, the WebDeployment media type does not support the Quick Build option because the .msi file is included in a .cab file.*

Performing Quick Builds

Task ***To perform a quick build:***

On the **Build** menu, click **Quick Build**.

Note ▪ *The SingleImage media type does not support the Quick Build option because the .msi file is included in the* `Setup.exe` *file. In addition, the WebDeployment media type does not support the Quick Build option because the .msi file is included in a .cab file.*

Command-Line Builds

You can build a release from the command line using `IsCmdBld.exe`. Building an installation in this way may be useful if you are trying to build from a batch file.

`IsCmdBld.exe` is located by default in the following location:

`InstallShield Program Files Folder``\System`

Do not move `IsCmdBld.exe` from its installed location.

If you pass parameters from the command line to `IsCmdBld.exe`, they override the settings for the release type.

Building from the Command Line

Task ***To build your installation from the command line:***

1. Open the Command Prompt window.
2. Change the directory to the following:

   `InstallShield Program Files Folder``\System`
3. Type the name of the command-line executable (**`IsCmdBld.exe`**) and the necessary parameters.

The following statement illustrates running `IsCmdBld.exe` to build a release:

```
IsCmdBld.exe -p "C:\InstallShield 2022 Projects\MyProject1.ise" -c COMP -e y
```

The first parameter, starting with `-p`, is the path to the InstallShield project file (.ise) that you want to build. The second parameter, `-c COMP`, specifies that you want your package to be compressed into one file. The final parameter, `-e y`, specifies that you want to include `Setup.exe` with the build.

Passing Command-Line Build Parameters in an .ini File

If you want to pass numerous parameters during a command-line build, or if you consistently pass the same parameters, it might be convenient to use an .ini file. The following statement illustrates running `ISCmdBld.exe` to build a release with parameters as specified in the **MySetup.ini** file.

```
ISCmdBld.exe -i "C:\InstallShield 2022 Projects\MySetup.ini"
```

You need to include the same information in the .ini file as you would if you were passing the parameters at the command line. There are four sections for this file:

- **[Project]**—In this section, include entries for the path to the project file (.ise), as well as the name of the product configuration. If you are building a patch, include an entry for the name of the patch configuration that you are building.
- **[Release]**—In this section, include entries for release configuration information such as the compression type (compressed or uncompressed), build flags, `Setup.exe` settings, and the release name.
- **[Mode]**—In this section, include any of the available optional entries, such as the Silent=yes entry if you want to build your release while suppressing any build errors or warning messages. This section also lets you indicate whether a log file should be created.
- **[BuildLocation]**—In this section, you can optionally specify the release output location.

Not all sections are required. As with passing parameters directly from the command line, parameters for requirements such as silent build and build location are optional. In the example .ini file below, these parameters are in the `[Mode]` and `[BuildLocation]` sections. You can omit these entries from your .ini file if you want to accept the defaults. By default, no log file is created, the installation is not run in silent mode, and your release is created in the project location that is specified on the File Locations tab of the Options dialog box.

Sample .ini File

The following tables contain sample entries from each of the four sections in a sample .ini file. The first column of each table shows a sample entry. The other columns provide the corresponding command-line parameter and description.

Entries in the [Project] Section

Table 4-2 ▪ Sample Entries in the [Project] Section of the .ini File

Entry	Corresponding ISCmdBld.exe Command-Line Parameter	Description
Name="C:\InstallShield 2022 Projects\Othello.ise"	-p	Path to the .ise file.
Product=Othello		Name of your product. This entry lets you override the value that is specified in the General Information view.

Entries in the [Release] Section

Table 4-3 ▪ Sample Entries in the [Release] Section of the .ini File

Entry	Corresponding ISCmdBld.exe Command-Line Parameter	Description
Configuration=COMP	-c	Compressed vs. uncompressed.
Name=Othello Beta	-r	Release name.
SetupEXE=yes	-e	Creates a `Setup.exe` file.

Entries in the [Mode] Section

Table 4-4 ▪ Sample Entries in the [Mode] Section of the .ini File

Entry	Corresponding ISCmdBld.exe Command-Line Parameter	Description
Silent=yes	-s	Runs in silent mode.
MergeModulePath="C:\..."	-o	Search path for merge modules (.msm files). For more information, see Specifying the Directories that Contain Merge Modules.

Table 4-4 ▪ Sample Entries in the [Mode] Section of the .ini File (cont.)

Entry	Corresponding ISCmdBld.exe Command-Line Parameter	Description
PrerequisitePath="C:\..."	-prqpath	Search path for InstallShield prerequisite files (.prq). For more information, see Specifying the Directories that Contain InstallShield Prerequisites.

Microsoft Build Engine (MSBuild)

InstallShield supports the Microsoft Build engine (MSBuild) included with the .NET Framework. MSBuild support enables you to build Visual Studio solutions with InstallShield projects in build lab environments where Visual Studio is not installed.

Overview

MSBuild is an extensible build framework designed to remove the build dependence on Visual Studio. The .NET Framework provides the capability to build projects or solutions from the command line or any other host of MSBuild. For more information on MSBuild, see the MSDN Library.

MSBuild Tasks

The flexibility and extensibility of MSBuild is controlled through atomic groupings of internal build steps referred to as *tasks*. One example of a task that ships with MSBuild is Csc, which can compile code from a Visual C# project. InstallShield installs an MSBuild task called InstallShield, which builds an InstallShield project, and a targets file that provides default build steps for the project. This customized task, along with the targets file, enables MSBuild to perform all required actions to build the InstallShield project as part of a Visual Studio solution.

This table describes the parameters of the InstallShield task.

Table 4-5 ▪ MSBuild InstallShield Task

Parameter	Type	Description
InstallShieldPath	String	This parameter specifies the path to folder containing the InstallShield application.
Project	String	This parameter specifies the location of the project file (.ise).
ProductConfiguration	String	This parameter specifies the product configuration for the release. The value for this parameter should be **Express**.

Table 4-5 ▪ MSBuild InstallShield Task (cont.)

Parameter	Type	Description
ReleaseConfiguration	String	This parameter specifies the release name. Valid options are as follows: • Custom • CD_ROM • DVD-10 • DVD-18 • DVD-5 • DVD-9 • SingleImage • WebDeployment
PatchConfiguration	String	This parameter specifies the name of the patch configuration that is being built in the Premier or Professional edition of InstallShield. ***Edition ▪*** *Patch configurations are supported in the Premier and Professional editions of InstallShield, but not the Express edition. Therefore, the PatchConfiguration parameter is not used with the Express edition.*
OutDir	String	This parameter specifies the fully qualified path to the folder where you want the output folders and files to be placed.
MergeModulePath	String[ ]	This parameter specifies one or more folders that contain the merge modules (.msm files) that are referenced by your project. InstallShield provides additional ways for specifying the folders that contain merge modules. For more information, see Specifying the Directories that Contain Merge Modules.
PrerequisitePath	String[ ]	This parameter specifies one or more semicolon-delimited folders that contain the InstallShield prerequisite files (.prq files) that are referenced by your project. InstallShield provides additional ways for specifying the folders that contain InstallShield prerequisite files. For more information, see Specifying the Directories that Contain InstallShield Prerequisites.

Table 4-5 ▪ MSBuild InstallShield Task (cont.)

Parameter	Type	Description
ReleaseFlags	String[ ]	This parameter enables you to specify any release flags that you want to include in your release. Separate multiple flags with commas. ***Edition ▪*** *Release flags are supported in the Premier and Professional editions of InstallShield, but not the Express edition. Therefore, the ReleaseFlags parameter is not used with the Express edition.*
PathVariables	ITaskItem[ ]	This parameter enables you to override the path variables for the project. Using this parameter, you can specify the path to the path variable and also define a value for the name of the path variable using the PathVariable subelement. To reference a file in a sibling project that is within the same Visual Studio solution folder, use the predefined path variable called VSSolutionFolder. For more information, see Using the VSSolutionFolder Path Variable with Visual Studio Solutions. For more information on MSBuild ITaskItem[ ] properties, see the MSDN Library. ***Edition ▪*** *This parameter applies to the Premier and Professional editions of InstallShield.*
PreprocessorDefines	ITaskItem[ ]	This parameter enables you to add or override the preprocessor defines for the project. Using this parameter, you can specify the definition of the preprocessor define and also define a value for the name of the preprocessor define using the Token subelement. For more information on MSBuild ITaskItem[ ] properties, see the MSDN Library. ***Edition ▪*** *This parameter applies to the Premier and Professional editions of InstallShield.*

Table 4-5 ▪ MSBuild InstallShield Task (cont.)

Parameter	Type	Description
OutputGroups	ITaskItem[ ]	This parameter specifies the project output groups from the Visual Studio solution. Using this parameter, you can specify the path to the source location of the project output group and also define these additional values using these subelements as listed: • **Name**—Name of the project • **OutputGroup**—Name of the project output group • **TargetPath**—Target path of the project output group (different from its source location) For more information on MSBuild ITaskItem[ ] properties, see the MSDN Library.
Build	String	This parameter specifies what type of build to perform. You can choose from these types of builds: • **Complete**—Specify this value if you want to generate a full build. • **Tables**—Specify this value if you have already performed a complete build and you simply want to rebuild only the .msi file part of the installation. This type of build is also known as a Quick Build. Note that this type of build cannot be performed with the SingleImage or WebDeployment types of release configurations. ***Edition ▪*** *Compile, TablesAndFiles, and UpgradeOnly are additional Build values that are available in the Premier and Professional editions of InstallShield.*
BuildCompressed	Boolean	This parameter specifies whether your release is compressed into one file or remains uncompressed in multiple files.
BuildSetupExe	Boolean	This parameter specifies whether you want to create a `Setup.exe` file along with your installation.
ProductVersion	String	This parameter specifies the product version. This is especially helpful if you want to increment the build version (the third field) of the product version. For information on valid product version numbers, see Specifying the Product Version.

Table 4-5 ▪ MSBuild InstallShield Task (cont.)

Parameter	Type	Description
PropertyOverrides	ITaskItem[ ]	This parameter enables you to override the value of a Windows Installer property or create the property if it does not exist. To use this parameter, include a property list of items whose value is the new property value, and whose metadata `Property` is the name of the property. This parameter is exposed as the InstallShieldPropertyOverrides ItemGroup passthrough to the PropertyOverrides property on the InstallShield task.
RunMsiValidator	String	This parameter enables you validate the .msi package using a .cub file. ***Edition ▪*** *This parameter applies to the Premier and Professional editions of InstallShield.*
RunUpgradeValidation	Boolean	This parameter specifies whether or not to run upgrade validation. ***Edition ▪*** *This parameter applies to the Premier and Professional editions of InstallShield.*
StopOnFirstError	Boolean	This parameter specifies whether or not to stop the build after an initial error.
MsiVersion	String	This parameter specifies the minimum version of Windows Installer that the installation can accept on the target machine.
DotNetFrameworkVersion	String	This parameter specifies the minimum version of the .NET Framework that the installation can accept on the target machine.
DotNetUtilPath	String	`Regasm.exe` and `InstallUtilLib.dll` are utilities that are included with each version of the .NET Framework. This parameter specifies the path for the directory that contains the 32-bit version of these files that you want to use at build time for releases that include .NET installer classes and COM interop. This is not the path to .NET Framework redistributable files.
Disk1Folder	Output String	This parameter specifies the location of the output folder.

Table 4-5 ▪ MSBuild InstallShield Task (cont.)

Parameter	Type	Description
SummaryInfoComments	String	Use this parameter to set **Summary Information Stream** comments at build time, such as including the build number. The comments that are added using this property can be viewed on the **Properties** dialog box of the built installer.
MSIPackageFileName	String	Use this parameter to specify the MSI package file name at build time. Specify the file name—without the period or the file extension—that InstallShield should use for the `.msi` file.

MSBuild Scripts

MSBuild understands one file format: its XML build script, which is used as the project file format for Visual C# and Visual Basic .NET projects (.csproj and .vbproj). MSBuild also has internal hooks to handle solution files and the Visual C++ project file format (.sln and .vcproj).

The InstallShield integration with Visual Studio uses an MSBuild-compatible XML format project file (.isproj), which enables MSBuild to seamlessly build Visual Studio solutions that include InstallShield projects.

Customizing the .isproj File

To incorporate changes for your InstallShield project into your .isproj file, add a PropertyGroup element or an ItemGroup element near the top of your .isproj file, or update an existing PropertyGroup or ItemGroup element. Then add InstallShield-related parameters as needed.

The following sample code from an .isproj file demonstrates how to do the following:

- Set the product version.
- Set the product name.
- Set a custom public property called `MY_PROPERTY` to the value *`My Value`*.
- Specify the following search paths for InstallShield prerequisites: `<ISProductFolder>\SetupPrerequisites` and `<ISProjectFolder>\MyCustomPrerequisites`.

```
<PropertyGroup>
   <InstallShieldProductVersion>1.2.3</InstallShieldProductVersion>
</PropertyGroup>
<ItemGroup>
   <InstallShieldPropertyOverrides Include="My New Product">
       <Property>ProductName</Property>
   </InstallShieldPropertyOverrides>
   <InstallShieldPropertyOverrides Include="My Value">
       <Property>MY_PROPERTY</Property>
   </InstallShieldPropertyOverrides>
   <InstallShieldPrerequisitePath Include="&lt;ISProductFolder&gt;\SetupPrerequisites"/>
   <InstallShieldPrerequisitePath Include="&lt;ISProjectFolder&gt;\MyCustomPrerequisites"/>
</ItemGroup>
```

Using MSBuild to Build a Release from the Command Line

Note ▪ *If you use MSBuild to build Visual Studio solutions with InstallShield projects, MSBuild requires .NET Framework 3.5 or later.*

MSBuild provides an easy way to build a release from the command line on a machine on which Visual Studio is not installed. The only components that you must have installed on the machine are the .NET Framework and InstallShield. Place a copy of your Visual Studio solution on the machine, and run MSBuild.

Task ***To use MSBuild from the command line:***

1. Open the Command Prompt window.
2. Change the directory to the one that contains `MSBuild.exe`:

   `C:\Windows\Microsoft.NET\Framework\`*`Version Folder`*`\`
3. Type the command-line statement to build the release build of the Visual Studio integration project. For example:

   ```
   MSBuild.exe C:\Folder Containing My Visual Studio Solution\My Solution.sln /
       property:Configuration=Release
   ```

Creating a Single Self-Extracting Installation File

Task ***To create a single self-extracting installation file:***

1. In the View List under **Prepare for Release**, click **Releases**.
2. In the **Builds** explorer, click **SingleImage**.
3. Click the **Build** tab.
4. For the **Compression** setting, select **Compressed**.
5. For the **Compress Media** setting, select **Yes**.
6. In the **Builds** explorer, right-click **SingleImage** and then click **Build**.

Any releases that you build with the SingleImage build type are compressed into one file.

Note ▪ *If you select Yes for the Setup Launcher setting, the file that InstallShield generates is an executable file (.exe). The executable file contains your Windows Installer database (.msi), as well as all of the files that are needed to install the Windows Installer (if appropriate) and your product. If you do not include a setup launcher, the single file that is generated is an .msi file that contains all of the logic and data that are necessary to install your product.*

Specifying the Required Execution Level for Your Setup Launcher on Windows Vista and Later Platforms

InstallShield lets you specify the minimum execution level required by your installation's `Setup.exe` file for running the installation (the setup launcher, any InstallShield prerequisites, and the .msi file) on Windows Vista and later platforms. You can configure this for each individual release in your project.

Task ***To specify the required execution level for a release:***

1. In the View List under **Prepare for Release**, click **Releases**.
2. In the **Releases** explorer, click the release that you would like to configure.
3. Set the **Required Execution Level** to the appropriate setting.
4. Click the **Setup.exe** tab.

The available options are:

- **Administrator**—`Setup.exe` requires administrative privileges to run. Administrators must provide consent, and non-administrators must provide credentials.
- **Highest available**—`Setup.exe` prefers administrative privileges. Administrators must provide consent to run it; non-administrators run it without administrative privileges.
- **Invoker**—`Setup.exe` does not require administrative privileges, and all users can run it without administrative privileges. `Setup.exe` does not display any UAC messages prompting for credentials or for consent. This is the default option.

If the Setup Launcher setting is set to Yes, InstallShield embeds a Windows application manifest in the `Setup.exe` launcher. This manifest specifies the selected execution level. Operating systems earlier than Windows Vista ignore the required execution level.

If the Setup Launcher setting is set to No, InstallShield does not embed the Windows application manifest in the `Setup.exe` launcher.

The benefit of elevating the required execution level is that privileges can be elevated only once if necessary to run `Setup.exe`, and that these privileges can be carried over to all of the installation's prerequisites and the .msi file without requiring multiple prompts for approval. Thus, if two of your prerequisites require administrative privileges, for example, you can change this setting to Administrator, and then end users are prompted only once during the installation, before Windows Installer runs the `Setup.exe` file. Note, however, that if you elevate the privileges and also launch the application at the end of the installation, the elevated privileges are carried over to the application. In most cases, running an application with elevated privileges on Windows Vista and later platforms is discouraged.

Note that an end user's installation experience is more secure when installations are run with only the permissions that they need. Unless an application is designed to be run only by system administrators, it should be run with the least privilege.

Preparing Your Installation for Internet Distribution

The way in which consumers receive software is rapidly changing. Before the advances in Internet technology and the introduction of high-speed connections, all software was shipped on some type of removable medium, such as floppy disks or CD-ROMs. Today, many people download their software directly from the Internet. In order to take advantage of this time- and money-saving software distribution process, you must package your installation in an easily downloadable and installable manner.

There are several criteria that your Web-ready installation may need to meet:

Compressed Size

Although many people now connect to the Internet through high-speed cable modems or DSL lines, others still use lower-speed modems. Package size is very important to people with slower connections due to the increased amount of online time required to download an application.

Self-Extracting

Many file compression utilities require a special client-side application to decompress the application files. This need for another utility complicates the download and installation process for users. To simplify the installation process, the compression utility you use should be self-extracting, so no other application is required.

Digitally Signed

To make your customers feel more secure about downloading and installing your software, you can digitally sign your application package. The digital signature identifies you and/or your company to end users and assures them that the application code has not been altered or tampered with since publication. To learn how to digitally sign your application, see Digital Signing and Security.

Easy to Use

Perhaps the most important aspect of packaging your installation for Internet distribution is making it easy to use. Your customers may not want to specify a location where the installation files should be saved and then search their computer to locate those files. Instead, the setup should be seamlessly integrated into the compression package, requiring only one step to begin the installation.

Proxy Server Support

You may want to configure your installation to download certain files only if they are needed on the target system. For example, the Windows Installer engine, the .NET Framework, and some InstallShield prerequisites may already be present on some or most target systems. Instead of embedding these files in your installation (which would increase your overall installation size), you can configure your project so that only the ones that are needed are downloaded at run time.

If your end users access the Internet through a proxy server and your installation is configured to download files, the installation uses the system proxy settings that are manually configured in Internet Explorer during the download. This occurs even if another browser on the target system is the default browser.

Note that InstallShield does not include support for the Automatically Detect Settings functionality in Internet Explorer. (If end users have the Automatically Detect Settings check box selected in Internet Explorer for their LAN connection and the installation needs to download files, the installation fails because the files cannot be downloaded. If it is possible that your end users may have the Automatically Detect Settings check box selected in Internet Explorer for their LAN connections, you may want to embed all of the files in your installation rather

than configure them to be downloaded; if the files are embedded, the failures can be avoided.) However, InstallShield does support the Automatic Configuration Script functionality that is set up for LAN connections in Internet Explorer.

Creating a Web-Deployable Release

Task ***To create a Web-deployable installation:***

1. Launch the Web Deployment Wizard by doing one of the following:
   - Click the **Web Deployment Wizard** button.
   - On the **Build** menu, click **Web Deployment Wizard**.
   - Press CTRL+W.
2. Provide the necessary information in the wizard panels.
3. In the **Summary** panel of the wizard, select the **Build the release after clicking Finish button** check box.
4. Click **Finish** to build the release.

You can distribute your release to a specified location through the Releases view. For more information, see Distributing Releases to a Folder or FTP Site Automatically.

Digital Signing and Security

You can digitally sign your installation and your application to assure end users that neither your installation nor the code within your application has been tampered with or altered since publication.

The Signing tab is where you specify the digital signature information—including the digital certificate files that a certification authority grated to you—that InstallShield should use to sign your files.

The Signing tab is also where you specify which files in your installation should be digitally signed by InstallShield at build time. InstallShield enables you to sign any and all of the following files in a release, depending on what type of project you are using:

- Windows Installer package (.msi file)
- `Setup.exe` file
- Any files in your release, including your application files

Windows Logo ▪ *All executable files (including .exe, .dll, .ocx, .sys, .cpl, .drv, and .scr files) in an installation must be digitally signed for the Windows logo program.*

To learn more about the various settings on the Signing tab, see Signing Tab.

Certification Authorities

A certification authority is an organization such as VeriSign that issues and manages digital certificates (also known as digital IDs). The certification authority validates the requester's identity according to prescribed criteria and issues a digital certificate. Obtaining a digital certificate requires providing the certificate authority with specific information about your company and your product.

For a list of certification authorities, see the Windows Root Certificate Program member list on the MSDN Web site.

SHA-1 vs. SHA-256 Certificates

InstallShield enables you to use digital certificates that use the SHA-256 or SHA-1 hashing algorithm for signing your installations and files at build time.

SHA-256 is favored over SHA-1, which is being deprecated because of the potential for security vulnerabilities. Microsoft announced that Windows will stop trusting items that were signed and timestamped with SHA-1 certificates after January 1, 2016. In addition, certification authorities—the organizations that issue certificates—are phasing out the creation of SHA-1 certificates. Thus, it is recommended that you replace any SHA-1 certificates in your InstallShield projects with SHA-256 certificates. For the latest information and more specific details, check with your certification authority.

If your project is configured to sign with a SHA-256 certificate, InstallShield uses a SHA-256 hash in the signature of the files that it signs at build time. If your project is configured to sign with a SHA-1 certificate, InstallShield uses a SHA-1 hash. Note that use of a SHA-1 certificate triggers build warning -7346 to alert you about the SHA-1 usage.

Using Certificate Files or Certificates in Certificate Stores to Generate a Digital Signature

When you are specifying the digital signature information that you want to use for signing your files and installations, InstallShield lets you choose between the following options:

- You can specify the .pfx certificate file on your machine that you want to use.
- You can reference a certificate store that contains the certificate that you want to use.

Option 1—.pfx File

You can use a personal information exchange file (.pfx) to digitally sign your installation and your application. The following tools enable you to create a .pfx file from a .pvk file and .spc file:

- `PVK2PFX.exe`—This is part of the Windows Platform SDK, and it is also included with Microsoft Visual Studio 2005.
- `pvkimprt.exe`—You can download this PVK Digital Certificate Files Importer tool from the downloads area on the Microsoft Web site (http://www.microsoft.com/downloads/details.aspx?FamilyID=F9992C94-B129-46BC-B240-414BDFF679A7&displaylang=EN).

The .pfx file is typically associated with a password.

Option 2—A Certificate in a Certificate Store

If you store your digital certificate in a certificate store, you can reference in your project the certificate store that contains the certificate that you want to use. If you use this method for signing, you will need to specify information such as the store name (Personal, Trusted Root Certification Authorities, Enterprise Trust, Intermediate Certification Authorities), the store location (user or machine), and the subject that identifies the specific certificate that you want to use.

If you configure your project to use a certificate that was imported with password protection into a store, Windows prompts for the password at build time when InstallShield is attempting to sign your project's files. The strong key protection that Windows uses does not permit InstallShield to provide the password to the cryptographic provider.

Adding a Timestamp to Digital Signatures

When you specify digital signature information for a release, InstallShield timestamps the digital signature at build time by default. A digital certificate's timestamp indicates the day and time when a file was signed; it also helps to prove that the certificate was not expired at the moment of signing. A timestamp from a trusted timestamp server typically extends the life of the digital signature beyond the certificate's validity period, protecting against certificate expiration.

To learn how to change the default timestamp server that InstallShield uses, or for information on how to disable timestamping, see Changing the Timestamp Server for Digital Signatures.

Digitally Signing a Release and Its Files at Build Time

InstallShield lets you configure digital signing settings for a release. At build time, InstallShield uses the settings that you have configured to sign your installation package, your `Setup.exe` file, and any other files in your release that meet the criteria that you have defined.

Task ***To configure digital signing for your release and its files:***

1. In the View List under **Prepare for Release**, click **Releases**.
2. In the **Builds** explorer, click the release that you want to sign.
3. Click the **Signing** tab.
4. Configure the following settings as appropriate:
   - **Signing Type**—Specify the method to digitally sign the build-generated files. Selecting the **Custom** option enables the **Path** and **Arguments** settings.
   - **Path**—Specify the sign tool's location to use that sign tool. To specify sign tool's location, click the ellipsis button (...) in this setting.
   - **Arguments**—Specify the argument for a sign tool's configuration.
   - **Certificate URL**—Type a fully qualified URL—for example, **http://www.mydomain.com**. This URL is used in your digital signature to link to a site that you would like end users to visit to learn more about your product, organization, or company.

- **Digital Certificate File**—Click the ellipsis button (**...**) in this setting. The Certificate Selection dialog box opens, enabling you to specify either the location of the .pfx file, certificate file (EV exported .cer file), or information about the certificate store that contains the certificate.
- **Certificate Password**—Note that if you configure your project to use a certificate that was imported with password protection into a store, Windows prompts for the password at build time when InstallShield is attempting to sign your project's files. The strong key protection that Windows uses does not permit InstallShield to provide the password to the cryptographic provider.
- **Sign Output Files—**Specify which files (`Setup.exe`, the .msi package, both of those files, or neither of those files) you want to be signed.
- **Use 64-Bit Signing**—Specify whether you want to use the 64-bit signing framework to digitally sign your package.
- **Signature Description**—Specify the signature description that you want to use for files that are specified in the Sign Output Files setting.
- **Sign Files in Package**—Specify whether you want to digitally sign all additional files, including InstallShield support files (English, language-independent, and advanced files) configured in the **Support Files** view, which are used only during the installation process.

  If you select **Yes**, use the other settings under the **Sign Files in Package** setting to indicate which files and file patterns should be signed and which should not be signed.

Note ▪ *Files and file patterns that should not be signed override any files and file patterns that should be signed. For example, if you specify* ****.exe*** *in an* ***Include*** *setting and in an* ***Exclude*** *setting, InstallShield does not sign any .exe files.*

Tip ▪ *For detailed information about any of the settings on the Signing tab, see Signing Tab.*

At build time, InstallShield signs the files as specified on the Signing tab. If the release is for an installation that includes merge modules, note that the files are signed before the merge module is merged.

Specifying the Run-Time Location for InstallShield Prerequisites at the Release Level

InstallShield lets you specify the run-time location of the InstallShield prerequisites that are included with your installation.

Task ***To specify where all of the InstallShield prerequisites should be located for a release:***

1. In the View List under **Prepare for Release**, click **Releases**.
2. Select the release that you want to configure.
3. Click the **Setup.exe** tab.
4. For the **InstallShield Prerequisites Location** setting, select the appropriate option.

For information about each of the available options, see Setup.exe Tab.

Task ***To specify different locations for each InstallShield prerequisite:***

1. In the **Redistributables** view, specify the appropriate location for each InstallShield prerequisite. For more information, see Specifying a Run-Time Location for a Specific InstallShield Prerequisite.
2. In the View List under **Prepare for Release**, click **Releases**.
3. Select the release that you want to configure.
4. Click the **Setup.exe** tab.
5. For the **InstallShield Prerequisites Location** setting, select **Follow Individual Selections**.

Note that if an InstallShield prerequisite is added to a project as a dependency of another prerequisite, the location for the prerequisite dependency follows the location setting of the prerequisite that requires it.

If you build a release that includes InstallShield prerequisites and both of the following are true, one or more build errors may be generated:

- You specify for the InstallShield prerequisites location that the prerequisites should be extracted from `Setup.exe` or copied from the source media (instead of being downloaded from the Web to the end user's computer).
- The prerequisite files are not on your computer.

To eliminate the build errors, remove the InstallShield prerequisite from your project, download the InstallShield prerequisite from the Internet to your computer, or change the InstallShield prerequisites location for the release to the download option; then rebuild the release.

Media Compression

Each release type (for example, CD_ROM, DVD, and SingleImage) that is available in the Releases view includes a setting called Compression. If you want to use a compression algorithm that will decrease the size of your installation, select Compressed for this setting. With this setting, your installation files are compressed into .cab files.

If you do not want your files to be compressed on the release media, select Uncompressed for the Compression setting.

Floppy Disk Distributions

Although it is not directly supported, you can distribute your release via floppy disk by using the Custom release type. The challenges that arise when you try to build your release on floppy disks are described below:

- **Disk Space**—If you want the Windows Installer to be installed before your product is installed, the Windows Installer must be installed first. Because of the file size for the Windows Installer redistributables, it is difficult to distribute your installation on a floppy disk if you need to distribute the Windows Installer engine as well.
- **Disk Spanning**—The .msi file that is created by InstallShield and that interacts with the Windows Installer service cannot be spanned across multiple disks. In addition, it must reside on the first disk of your

installation. Therefore, if you want to include all of your files in one compressed .msi file, it might not fit on one floppy disk. However, if your files can remain uncompressed, they can be included on successive disks of the installation.

Building a Release for Floppy Disk Distribution

Task ***To build a release for floppy disk distribution:***

1. In the View List under **Prepare for Release**, click **Releases**.
2. In the **Builds** explorer, click **Custom**.
3. Click the **Build** tab.
4. For the **Media Size** setting, type **`1.4`**.
5. For the **Media Size Unit** setting, type **`MB`**.
6. In the **Cluster Size** setting, type the cluster size (in bytes) according to the disk capacity: **`512`** or **`1024`**.
7. For the **Compression** setting, select **`No`**.
8. For the **Generate Autorun.inf File** setting, select **`No`**.
9. Click the **Setup.exe** tab.
10. For the **Setup Launcher** setting, select **`No`**.
11. In the **Builds** explorer, right-click **Custom** and then click **Build**.

Caution ▪ *If you build a multiple-disk installation, you need to set the volume label on the second and subsequent disks. The volume labels must be DISK2 (for the second disk), DISK3 (for the third disk), and so on. For more information, see* *Setting Volume Labels**.*

Setting Volume Labels

Task ***To set the volume label for a CD-ROM or DVD-ROM:***

Consult the requirements of your CD-ROM or DVD-ROM creation software.

Task ***To set the volume label for a floppy disk:***

1. Open Windows Explorer.
2. Right-click the drive containing the floppy disk and then click **Properties**.
3. Click the **General** tab.
4. In the **Label** box, type the appropriate volume label.

Tip ▪ *If you build a multiple-disk installation, you need to specify volume labels for the second and subsequent disks. The manner in which you do this depends upon the distribution media you use. The volume labels must be DISK2 (for the second disk), DISK3 (for the third disk), and so on.*

Creating a Windows Installer Package (.msi) as Output

You can generate an .msi file to package your installation.

Task ***To create an .msi file as your InstallShield output:***

1. Create your installation, including the necessary files, features, registry entries, and shortcuts.
2. In the View List under **Prepare for Release**, click **Releases**.
3. In the **Builds** explorer, click the type of media that you want to build.

   Note ▪ *You cannot generate an .msi file accessible to your end users through the WebDeployment release type or the SingleImage release type.*
4. Click the **Build** tab.
5. For the **Compression** setting, select **Uncompressed**. This instructs InstallShield to not embed the .msi file into the `Setup.exe` file.
6. Click the **Build** button.

InstallShield builds your release and stores it in the `<Project Location>\<Project Name>\Express\<Release Type>\DiskImages\Disk1` folder. The .msi file is in this folder.

Autorun

Sometimes when you insert software CDs or DVDs into your machine's drive, a multimedia browser automatically launches, enabling you to easily install the software. This autorun functionality is accomplished using a text file (`Autorun.inf`) on the root level of the CD or DVD. Primarily, this file is used to launch either a CD or DVD browser or an installation.

Enabling Autorun for Your CD or DVD

Task ***To enable your CD or DVD to autorun when it is inserted into the target system's CD or DVD drive:***

1. Create a text file called **`Autorun.inf`**.
2. In the `Autorun.inf` file, specify the name of the file to automatically launch when an end user inserts the CD or DVD into the drive. Use the following syntax:

   ```
   [autorun]
   ```

```
open = filename
```

where **filename** is the name of the file that you are launching. For example, if you are including `Setup.exe` in your installation, enter **`Setup.exe`** in the place of **filename**.

3. Place the `Autorun.inf` file on the root level of your CD or DVD.

Testing and Running Installations

Before distributing your installation, you should test run it to ensure you discover any issues rather than your customers. InstallShield enables you to test the end-user dialogs (without copying any files to the target system) and run the entire installation including transferring files.

Testing and Running Installations

Testing your release is a necessary step in the installation development process. You can test your release in one of two ways:

- **Run your installation**—This option executes your installation exactly as it would on an end user's machine. All files are transferred, shortcuts and registry entries are made, and the end-user interface is displayed. Although you can run this type of test from the InstallShield interface, it is also a good idea to test your installation on several clean machines before you release it to the public.
- **Test your installation**—When you test your installation, only the end-user interface elements are executed. No file transfer takes place and no changes to your machine occur. The only exception to this rule is if you have custom actions in your installation. All custom actions execute during this test.

Task ***To test or run an installation:***

1. In the View List under **Prepare for Release**, click **Releases**.
2. In the **Builds** explorer, click the release that you want to test or run.
3. Click the **Run** button or the **Test User Interface** button on the toolbar.

Running an Installation in Silent Mode

If you do not want to have the `Setup.exe` progress bar displayed when it launches, you can use the `/s` command-line parameter. For example, if you enter **`Setup.exe /s`**, `Setup.exe` launches, but the user interface is not displayed. If you want the .msi package to run silently as well, you need to pass the `/qn` command-line parameter through `Setup.exe` using the `/v` parameter—for example:

```
Setup.exe /s /v/qn
```

Conditional Installations

InstallShield provides the capability to make your installation conditional based on several criteria. This can be accomplished using the Requirements view. If the installation is run on a machine that does not meet the specified conditions, the installation exits and your application is not installed.

Distributing Installations

Once you have created your installation, you may need to distribute it to a specified location. This location can be a network drive, a CD, a different location on a local drive, or an FTP site. When you distribute your installation, the disk image that was created when you built your installation is copied to the location that you specify.

Distributing Releases to a Folder or FTP Site Automatically

When your release is built and tested, the only remaining task is to distribute it to the appropriate location. You can manually copy your release to the appropriate location, or you can use the Events tab in the Releases view to configure the release so that InstallShield automatically copies the release to the appropriate location—a local or network location, or an FTP site.

Task ***To configure InstallShield to automatically distribute your release to a particular location:***

1. In the View List under **Prepare for Release**, click **Releases**.
2. In the **Releases** explorer, select the release that you want to configure.
3. Click the **Events** tab.
4. Configure the settings as appropriate. To learn more about the settings on the Events tab, see Events Tab.

Note ▪ *If your installation consists of only one disk, the contents of the Disk1 folder are copied to the release location, but not the folder itself. If your installation spans across multiple disks, the folders and their contents are copied to the release location.*

If you want the build engine to copy your release to the specified location after every build, select Yes for the Distribute After Build setting.

5

Updating Applications

Installing upgrades for applications is a much more common operation than installing the original release of an application. This makes creating effective, reliable upgrades a very important task.

“Updating Applications” provides solid background information on the different types of upgrades and also clears up common misconceptions about patching. It helps you determine the best upgrade solution for your product and guides you through the steps of creating upgrades and patches. In addition, this section explains how you can use FlexNet Connect to notify end users that a new version of your product is ready for release.

Upgrades Overview

The cost of maintaining software applications is often more expensive than the cost of original development. This makes creating effective, reliable upgrades an important task. Being able to distribute robust upgrades for an application depends on how the original installation package was structured and distributed.

Windows Installer provides different methods for implementing three different upgrade types: small update, minor upgrade, and major upgrade. Upgrades can be packaged as either a full installation or a patch. A patch is simply another mechanism for implementing a type of upgrade. With a patch, however, you deliver to your customers only the bits required to change an installed file into a new file, unlike a full release.

Major Upgrades

Project ▪ *This information applies to Express projects.*

In a major upgrade, the product changes are large enough to merit changes to both the product version number and the product code, as well as the package code. An example is updating version 1.2 of a product to 2.0. A major upgrade acts like a first-time installation if an earlier version is not present. If an earlier version is present, a major upgrade typically uninstalls the earlier version and then installs the new version.

How Major Upgrades Work

As the Windows Installer help implies, you must change the product code for the latest version of your installation in order to perform a major upgrade.

Note ▪ *You can set the product code for your installation in the General Information view. It does not matter what the new product code is, as long as it is unique.*

Once you update the product code for your latest installation, you need to use the Upgrade Paths view to specify information for any previous version or versions that you want to upgrade. You can do this in the latest installation project for your product.

For more information about what is involved in creating an upgrade, see Creating Full-Installation Upgrades.

Major Upgrades at Run Time

When an earlier version of the product is not available on the target machine and the end user runs a major upgrade, the installation behaves as a first-time installation.

If an earlier version of the product does exist on the target machine and the end user runs a major upgrade, they will encounter almost the exact same installation experience as if they were to install the latest application on a clean target machine. The difference is that before installing its new resources, the installation will first remove the old application version and its resources from the target machine. This removal is reflected in the progress bar of the Setup Progress dialog, which gives end users the ability to see the progress of the uninstallation. After the removal of the previous installation has completed, the resources from the latest installation will then be installed onto the target machine.

This type of upgrade is a complete uninstallation and then reinstallation of all of the resources associated with your application. Therefore, any data for your application that has been configured by the end user may be completely deleted from the end user's machine. If you need to retain some of the end-user data, you will need to create a custom action that backs up this data, and then replaces it after the installation of new data has completed.

Minor Upgrades

Edition ▪ *You can create a minor upgrade packaged as a full installation in InstallShield Premier Edition or InstallShield Professional Edition. You can also create a minor upgrade packaged as a QuickPatch project in InstallShield Premier Edition, InstallShield Professional Edition, or InstallShield Express Edition.*

A minor upgrade is a change to the product database and files large enough to merit a change to the `ProductVersion` property but not to the `ProductCode` property. In other words, both the package code and the product version number are different than those in the earlier installation package, but the product code of the minor upgrade does not change. An example is updating version 1.1 of a product to version 1.2. Minor upgrades usually do not have significant changes to the installation organization between versions. A minor upgrade packaged as a full installation acts like a first-time installation if an earlier version is not present, or it installs over an existing installation of a product. Essentially for an upgrade of an existing installation, a minor upgrade installs the differences between your version 1.2 and 1.1 application.

Small Updates

Edition ▪ *You can create small updates in InstallShield Premier Edition or InstallShield Professional Edition.*

In essence, a small update is a type of upgrade used to modify a few files for an installed application; it typically delivers small bug fixes. A small update consists of product changes (such as hotfixes) so small that no change to the product version is necessary or desired. A package code change is required for a small update.

A drawback to small updates installed with versions of Windows Installer earlier than 3.0 is that external programs, including installers for later versions of your product, cannot distinguish between the original version and the updated version. In addition, Windows Installer versions earlier than 3.0 cannot enforce the correct order in which small updates should be applied.

Patching

Edition ▪ *You can create standard patches in InstallShield Premier Edition or InstallShield Professional Edition.*

Patching is a streamlined mechanism for updating earlier versions of a Windows Installer installation package, thereby updating the application. With a patch, you deliver to your customers only the bits required to change an installed file into a new file. One benefit of a patch is that the size of the upgrade package can be significantly smaller than the full-installation package required to deliver the same upgrade. Keeping an upgrade package as small as possible allows you to more easily deliver your upgrades over the Internet.

However, you should note that a patch may not always present the best solution. For example, if you want to change your installation from compressed to uncompressed, or vice versa, you should package your upgrade as a full installation, but not as a patch. To learn more about determining the best packaging option for your upgrade, see Packaging Options for Upgrades.

A patch is delivered in the form of a patch package (.msp) file, which your end user can apply to an installed product. A patch package is capable of updating as many earlier versions of an installation package as required. A patch package contains separate transforms and instructions for updating each previous version that you specify.

An important aspect of patch creation is generating a patch creation properties (.pcp) file, which defines the parameters on which the patch package is to be created. The .pcp file is a database that has a specific schema.

QuickPatch Projects

Project ▪ *You can create a QuickPatch to update an earlier version of your product if the installation for the earlier version was created with either of the following project types:*

- *Express*
- *QuickPatch*

A QuickPatch project is a specific type of project recommended for installation authors who want to ship small, single updates to their users. Changes that are more extensive such as adding custom actions and changing .ini data typically require a standard patch.

QuickPatch authoring provides a simple alternative to creating a standard patch (which is available in InstallShield Premier Edition and InstallShield Professional Edition), even though it provides less customization. Fundamentally, both patch creation methods produce the same deliverable types: .msp and .exe files.

With a QuickPatch, you can do any of the following:

- Add new files to the original installation or an earlier QuickPatch.
- Delete files in the original installation.
- Delete files that were added with an earlier QuickPatch.
- Perform the same set of above operations on registry entries.
- Remove custom actions that were included with the original installation but that do not apply to the current QuickPatch project.

The creation of a QuickPatch project always begins with the Create New QuickPatch Wizard. Completing the wizard ensures that all basic requirements for the QuickPatch project are met. Then you can configure project settings once the QuickPatch project opens in InstallShield.

Determining the Best Upgrade Solution

The first step in creating an installation for any type of upgrade is identifying whether you want to address target systems that do not have any earlier version of your product. Then you can determine what type of method you should use to package the upgrade. The following table presents a general overview to help you decide which method you should use. For a more in-depth discussion of the techniques for packaging the upgrade, see Packaging Options for Upgrades.

Table 5-1 ▪ Possible Upgrade Solutions for Express Projects

Status of the Target Systems	Type of Installation Needed	Technique for Packaging the Update
Some of the target systems have an earlier version of the product, and some do not have any version of the product.	If file size is not an issue, you can create one installation that does both of the following: ● Behaves as a first-time installation if an earlier version of the product is not already present on the target system ● Updates an existing product if it is already installed on the target system	Create a full-installation upgrade. For more information, see Creating Full-Installation Upgrades.

Table 5-1 ▪ Possible Upgrade Solutions for Express Projects (cont.)

Status of the Target Systems	Type of Installation Needed	Technique for Packaging the Update
Some of the target systems have an earlier version of the product, and some do not have any version of the product.	If you want a small installation for end users who need to update an earlier version of the product, you can create two separate installations: ● A full installation that behaves as a first-time installation. ● A smaller installation that updates one or more earlier versions of an already installed product. Since this installation consists of only the changed data between the versions to be updated, it usually enables you to deploy your upgrade using much less bandwidth than that required to deploy a full-installation package.	For the first-time installation, create a full-installation upgrade. For more information, see Creating Full-Installation Upgrades. For end users who have an earlier version of your product, you might be able to create a QuickPatch project instead of a full-installation upgrade. To determine if a QuickPatch project is suitable, see Packaging Options for Upgrades.
All of the target systems have an earlier version of the product.	You can create a small installation that updates one or more earlier versions of an already installed product. Since this installation consists of only the changed data between the versions to be updated, it usually enables you to deploy your upgrade using much less bandwidth than that required to deploy a full-installation package.	You might be able to create a QuickPatch project instead of a full-installation upgrade. To determine if a QuickPatch project is suitable, see Packaging Options for Upgrades.

Packaging Options for Upgrades

Project ▪ *This information applies to the following project types:*

- *Express*
- *QuickPatch*

When you want to prepare an installation that updates a version of your product installed on an end user's machine, you can consider two different methods for packaging your upgrade:

- You can package your upgrade as a full installation that updates an existing product if an earlier version is installed, or behaves as a first-time installation if no earlier version is present.
- You can package your upgrade as a QuickPatch that contains only the changed data (.msi data and byte-level file differences) between the versions to be updated.

Full-Installation Packages

A major upgrade acts like a first-time installation if an earlier version of the product is not present. If an earlier version is present, a major upgrade typically uninstalls the earlier version and then installs the new version.

QuickPatch Packages

QuickPatch packages enable you to distribute just the bits and portions of the database necessary to update your application's files to a specific version, possibly resulting in a much smaller package than an upgrade packaged as a full installation. This enables you to deploy your upgrades using much less bandwidth than that required to deploy a full-installation package.

***Note** ▪ A patch is not a type of upgrade. Patching is simply a mechanism for distributing an upgrade with a small footprint.*

Determining the Best Packaging Option for Your Upgrade

The topic Determining the Best Upgrade Solution includes a table that you can review to determine what type of packing option you should use to update an earlier version of your product. In some cases, a QuickPatch may be seem to be the ideal mechanism for packaging your upgrade. However, under certain conditions, you should package your upgrade as a full installation instead of a QuickPatch. The following table serves as a guide to help you determine which type of upgrade best suits your needs. If any one of the requirements for your upgrade is not appropriate for a QuickPatch, you should create a full-installation upgrade.

Table 5-2 ▪ Full-Installation Upgrade vs. QuickPatch

Requirement for the Upgrade or QuickPatch	Create a Full-Installation Upgrade?	Create a QuickPatch?	Note
Ability to apply many cumulative updates that upgrade a base package	Yes	Yes, if you are using QuickPatch streamlining	If you are not using QuickPatch streamlining, you cannot patch more than 15 times. For more details, see Specifying Whether to Streamline the QuickPatch Package.
Change the name of the .msi package	Yes	No	The default file name is taken from the Product Name property, provided the .msi file is not compressed in a `Setup.exe` installation launcher.
Enable end users to install earlier versions and the latest version on the same machine	Yes	No	
Add a new subfeature	Yes	No	

Table 5-2 ▪ Full-Installation Upgrade vs. QuickPatch (cont.)

Requirement for the Upgrade or QuickPatch	Create a Full-Installation Upgrade?	Create a QuickPatch?	Note
Add, modify, or remove a file	Yes	Yes	
Add, modify, or delete registry data	Yes	Yes	All new registry data being added with a QuickPatch must be associated with a feature that already exists in the original product.
Add, modify, or delete a shortcut	Yes	No	
Add, modify, or delete custom actions	Yes	Can only delete custom actions that were included in the original base installation.	
Add or remove a redistributable	Yes	No	
Add, modify, or remove ODBC resources	Yes	No	
Edit an .ini file	Yes	No	
Configure server settings such as IIS Web sites and component services	Yes	No	

Also note the following details when you are determining what type of packing option you should use to update an earlier version of your product:

- If the target image was created with Windows Installer 1.2 or earlier, and the upgraded image is created with Windows Installer 2.0 or later, you should package your upgrade as a full installation, but not as a QuickPatch. Creating a QuickPatch for a package that spans this schema inflection point can be problematic.
- If you want your upgrade to move one or more files on the target system from one location to another, you should package your upgrade as a full installation, but not as a QuickPatch. If your end users install a patch for an upgrade that moves files on the target system, problems may occur. For example, the patch may not work, a repair to the target system may not work, a subsequent patch may not work, or end users may not be able to uninstall the product. As a workaround, you can create your QuickPatch so that it deletes the files in the old location and adds the files to the new location.
- If you want to change your installation from compressed to uncompressed, or vice versa, you should package your upgrade as a full installation, but not as a QuickPatch. If you use a QuickPatch in this scenario, a repair

to the target system may not work, a subsequent QuickPatch may not work, or the end user may not be able to uninstall the product.

- If you need to move files from one .cab file to another, or if you need to change the order of files in a .cab file, you should package your upgrade as a full installation, but not as a QuickPatch.
- If your original installation had more than 32,767 files but your latest installation has fewer than 32,767 files, a QuickPatch will fail. Similarly, if your original installation had fewer than 32,767 files but your latest installation has more than 32,767 files, a QuickPatch will fail. In either case, you should package your upgrade as a full installation.

  Note that if both the original installation and the latest installation have more than 32,767 (or both have fewer than 32,767 files), you can package your upgrade as a QuickPatch.

Working with Upgrades and QuickPatch Projects

If you are going to package your upgrade as a full installation, you begin by opening the latest version of your installation project and making the necessary changes, such as adding files and registry entries, as needed.

If you would like to package your upgrade as a QuickPatch, you begin by creating a new project—a QuickPatch project. With a QuickPatch project, you identify which earlier releases should be patched by your QuickPatch.

***Edition** ▪ InstallShield Express Edition enables you to create major upgrades as full-installation packages that remove an earlier version of the product if it is present on the target machine before installing the new version. InstallShield Express Edition also enables you to create minor upgrades packaged as QuickPatch packages. If you need to create small updates or standard patches, consider upgrading to InstallShield Premier Edition or InstallShield Professional Edition.*

Refer to the topics in this section to learn how to create upgrades and QuickPatch projects.

Understanding File Overwrite Rules

The Windows Installer service uses several file overwrite rules, by default, to determine whether a file included in an upgrade should overwrite a file that already exists on the target system. The rules apply when the `REINSTALLMODE` property uses the `o` setting to install over older files on the target system. To change this behavior, you can replace the `o` option with one of the following values:

- `p`—Reinstall only if there is no equivalent file on the target system.
- `e`—Reinstall if the file is missing or if it is an earlier or equal version.
- `d`—Reinstall if file is missing or different.
- `a`—Reinstall all files, regardless of version.

Note that the setting for `REINSTALLMODE` applies to all of the features being installed; it cannot be set for an individual feature. In addition, setting `REINSTALLMODE` to include `a` will likely cause prompts for the original installation source during the application of a patch.

Updating the Package Code, the Product Version, and the Product Code

Several Windows Installer codes help identify a product:

- **Package Code—**Part of the Summary Information Stream, the package code identifies a particular database. The package code is not a Windows Installer property. Any two .msi databases with identical package codes must have identical contents. Therefore, you should change the package code for each build.
- **ProductVersion—**This is a Windows Installer property that contains the product version. Note that Windows Installer uses only the first three fields of the `ProductVersion` property for version comparisons. For example, for a product version of 1.2.3.4, the 4 is ignored. (Note that this is true for comparisons of `ProductVersion` values, and not for file versions.)
- **ProductCode—**This is a Windows Installer property that contains the GUID of a product. Windows Installer treats two products with different `ProductCode` GUIDs as unrelated, even if the values for the `ProductName` property are the same.
- **UpgradeCode—**This is a Windows Installer property that contains a GUID representing the product family. The `UpgradeCode` should be consistent across different versions and languages of a family of related products for patching purposes. You can set the `UpgradeCode` for an upgrade in the Upgrade Paths view.

For any type of upgrade, you must change various combinations of the package code, product version, and product code in the General Information view to identify the product being installed. The upgrade code must be the same for all versions of the product. The following table identifies when each code should be changed for different types of upgrades.

Table 5-3 ▪ Codes that Need to Be Changed for the Different Types of Upgrades

	Package Code	Product Version	Product Code	Upgrade Code
Small Update	X			
Minor Upgrade	X	X		
Major Upgrade	X	X	X	

Creating Full-Installation Upgrades

Project *▪ This information applies to Express projects.*

Once you have determined that a full-installation upgrade is the best upgrade solution for you, you can begin to create your upgrade in the Upgrade Paths view. You can start by creating a new Express project, or you can open the installation for the latest version of your product and modify it as needed.

Note that a major upgrade signifies significant change in functionality and warrants a change in the product code and the product version; you can update these values in the General Information view.

Note ▪ *When you change the product code, Windows Installer treats your latest and previous product versions as unrelated, even though the* `ProductName` *values are likely the same. If you want both versions of your product to be installable on the same system, you can simply change the product code and the main installation directory (often INSTALLDIR).*

Upgrade paths are for major upgrades only. The Upgrade Paths view is only for upgrading an entire installation, and not for upgrading a few files (patching). If you need patching capabilities, you can create a QuickPatch project.

If you do not use the Upgrade Paths view to create an upgrade for an earlier version of the product, your end users may need to manually uninstall the earlier version of your product before installing the new version. The Upgrade Paths view can handle both the uninstallation of one or more products, as well as the installation of the new version. For example, if your users have both 1.0 and 2.0 of your product installed on their system, and you release 3.0, you can use the Upgrade Paths view to remove any or both of the earlier versions by adding two entries in the Upgrade Paths explorer. Each upgrade path must have a separate entry in the Upgrade Paths explorer.

The following table shows the relationship between the product code and the upgrade code through a first installation version and two product updates. If the upgrade paths were set up based on this information, version 1.0 would upgrade to either version 1.1 or 2.0; version 1.1 would upgrade to 2.0.

Table 5-4 ▪ Sample Codes for Upgrades

Product Version	Product Code	Upgrade Code
1.0	{D73FFD24-6087-4D5A-82AB-1F6B1A680433}	{A00700B1-2345-6789-ABCD-EF0123456789}
1.1	{D73FFD24-6087-4D5A-82AB-1F6B1A680433}	{A00700B1-2345-6789-ABCD-EF0123456789}
2.0	{A1071CCE-11A1-11A2-B1A2-B3000D067A34}	{A00700B1-2345-6789-ABCD-EF0123456789}

For more information on creating an upgrade using InstallShield, refer to this section of the InstallShield Help Library.

Preventing the Removal of Assemblies from the Global Assembly Cache During Upgrades

Project ▪ *This information applies to Express projects.*

By default, upgrades that are created in the Upgrade Paths view are configured to remove the earlier version of the product before installing the new version. That is, the RemoveExistingProducts action is scheduled before the InstallFinalize action. This default sequencing behavior may cause a problem if the product includes an assembly

that is installed to the global assembly cache (GAC): after the upgrade is applied, the assembly may be missing from the GAC. The issue occurs because Windows Installer cannot properly reference count the assembly; therefore, it is removed but not reinstalled during the upgrade.

To work around this Windows Installer issue, you can configure your project in InstallShield so that the new version of your product is installed before the earlier version is removed.

Task ***To configure your project so that upgrades are installed before earlier versions of your product are removed, and to prevent the removal of the assemblies from the GAC:***

1. In the View List under **Organize Your Setup**, click **Upgrade Paths**.
2. In the **Upgrade Paths** pane, click **Upgrade Paths**.
3. In the help pane on the right, click the **Resequence RemoveExistingProducts** button.

InstallShield reschedules the RemoveExistingProducts action so that it occurs after the InstallFinalize action.

Tip ▪ *To revert back to the default behavior, click the Resequence RemoveExistingProducts button again.*

To learn more about the Windows Installer issue, see Microsoft Knowledge Base article 905238.

Adding an Upgrade Item

Project ▪ *This information applies to Express projects.*

If you have released earlier versions of your product and you want to ensure that end users are able to upgrade to the current version without manually uninstalling the earlier version and then also installing the current version, use the Upgrade Paths view to indicate upgrade information.

Task ***To add an upgrade item:***

1. Open the **Upgrade Paths** view.
2. Right-click the **Upgrade Paths** explorer and click **New Upgrade Path**. The **Open** dialog box opens.
3. Do one of the following:
   - If the installation for the version that you want to upgrade exists on your system, browse to the .msi file or the .exe file and then click **Open**. InstallShield copies the upgrade code from that earlier version for your current project.
   - If the installation for the version that you want to upgrade is not on your system, click **Cancel**.

InstallShield adds a new upgrade item.

Configuring Upgrade Properties

Project ▪ *This information applies to Express projects.*

When you add an upgrade item in the Upgrade Paths view, there are several settings that you can configure. To learn about each of those settings, see Upgrade Path Settings.

Removing Upgrade Items

Project ▪ *This information applies to Express projects.*

Task ***To remove an upgrade item from the Upgrade Paths view:***

1. Open the **Upgrade Paths** view.
2. In the **Upgrade Paths** explorer, right-click the upgrade item and click **Delete**.

Patching Considerations

Project ▪ *This information applies to QuickPatch projects.*

Following are some guidelines for creating QuickPatch projects.

Patches for Compressed Installations

The patch creation process requires an uncompressed release of the previous installations and the latest installation. If the installations are compressed installations, you can use an administrative image of the releases.

InstallShield automatically creates an administrative image for you if you specify an uncompressed image in a QuickPatch project.

Creating an Update Launcher (Update.exe)

InstallShield lets you specify whether you want your update to include an `Update.exe` update launcher. An `Update.exe` update launcher is required in the following cases:

- You want to automatically update or install the Windows Installer engine on a target system, when necessary.
- You want to automatically install the .NET Framework on a target system, when necessary.

The `Update.exe` update launcher is a bootstrap application that manages the aforementioned scenarios.

If you configure InstallShield so that it does not create an `Update.exe` update launcher, it creates an .msp file.

For more information, see Specifying Whether to Build an Update.exe Update Launcher for a QuickPatch Package.

Windows Installer 3.0 (and Later) and Patches

Windows Installer 3.0 and later includes many patch-related enhancements. If your patch or QuickPatch will be applied to target machines with Windows Installer 3.0 or later, you can take advantage of those enhancements. For more information, see the following:

- Patch Sequencing
- Patch Uninstallation
- Non-Administrator Patches

Patch Sequencing

Project ▪ *This information applies to QuickPatch projects.*

InstallShield enables you to specify the order that Windows Installer version 3.0 or later should apply small-update patches to an installed product, regardless of the order in which they are provided to the target machine. With patch sequencing data, you can ensure that Windows Installer knows the intended relationships among the upgrades packaged within a family of patches. Consequently, applying patch 1 of a product after patch 2 has already been applied will register patch 1 without overwriting patch 2 files. For versions of Windows Installer earlier than version 3.0, the patch sequence is ignored, and any small-update patches are applied to the product in the order that they are provided to the target machine.

The patch sequencing functionality available with Windows Installer 3.0 and later simplifies the patch creation process. The following sections show how.

Creating Patches to Be Applied with Versions of Windows Installer Earlier than 3.0

If you need to create your patches so that they can be applied to your product via versions of Windows Installer earlier than 3.0, it is recommended that you avoid creating small updates. Small updates do not change the product version; therefore, external programs, including installers for later versions of your product, cannot distinguish a product with the small update applied from one without the small update. For scenarios limited to versions of Windows Installer earlier than 3.0, you need to plan around such limitations of the installer, thus targeting an increasing number of possible earlier product states. The sample application lifecycle presented in the following table illustrates the resulting complexity.

Table 5-5 ▪ Sample Application Lifecycle for Patches Applied with Versions of Windows Installer Earlier than 3.0

Application Package	Product Version	Previous Setups Targeted by Package
1. Base installation	1.0	—
2. Minor upgrade	1.1	1.0
3. Minor upgrade	1.2	1.0, 1.1
4. Minor upgrade	1.3	1.0, 1.1, 1.2
5. Minor upgrade	1.4	1.0, 1.1, 1.2, 1.3

Table 5-5 ▪ Sample Application Lifecycle for Patches Applied with Versions of Windows Installer Earlier than 3.0

Application Package	Product Version	Previous Setups Targeted by Package
6. Major upgrade	2.0	1.0, 1.1, 1.2, 1.3, 1.4

Creating Patches to Be Applied with Windows Installer 3.0

With the patch sequencing functionality available with Windows Installer 3.0 and later, you can safely use a small-update patch to deliver a discrete upgrade for several different versions of a product, even though small updates do not change the product version. Unlike a small update, a minor upgrade changes the product version. Minor upgrades also form the framework for the sequencing of small-updates patches. If a small update for version 1.1 of a product is applied to version 1.2, the installer registers the small update on the target system and applies it as if it were encountered before the 1.2 minor upgrade was applied.

Small-update patches also enable Windows Installer 3.0 and later to maintain a valid product state as other patches are applied and removed individually from the product. In addition, patch sequencing lets you generate upgrade packages from a smaller set of earlier product states without requiring you to consider every possible combination of patches that could exist on the target machine. The sample application lifecycle presented in the following table illustrates this advantage.

Table 5-6 ▪ Sample Application Lifecycle for Patches Applied with Windows Installer 3.0

Application Package	Patch Sequence Number	Product Version	Previous Setups Targeted by Package
1. Base installation	—	1.0	—
2. Small update	1	1.0	1.0
3. Small update	2	1.0	1.0
4. Small update	3	1.0	1.0
5. Small update	4	1.0	1.0
6. Minor upgrade	—	1.1	1.0

All of the small updates in the table above belong to the same patch family. Windows Installer 3.0 and later uses the patch family to compare a small-update patch with all of the other patches within the same family and determine the order in which each of the patches should be applied to the target machine. Patch sequences are added to the `MsiPatchSequence` table of the patch package database. This table defines the relationships between patches that target the same family of patches.

Patch Uninstallation

Project ▪ *This information applies to QuickPatch projects.*

Windows Installer 3.0 or later supports the uninstallation of patches for small updates and minor upgrades; the patches to be uninstalled must have been installed with Windows Installer 3.0 or later. When a patch is uninstalled, the product is returned to the state it was in before the patch was uninstalled. With earlier versions of Windows Installer, an end user who wants to remove a patch needs to uninstall the patched product and then reinstall the product without applying the patch. In this case, any patches that should not be removed must be reapplied.

End users can uninstall patches through Add or Remove Programs on systems running Windows XP SP2 and later. For systems running Windows Installer 3.0 or later with earlier versions of Windows, patches must be uninstalled from the command line. For more information, see Uninstalling Patches in the Windows Installer Help Library.

The uninstallation of patches is not enabled by default because not all patches can be uninstalled. For example, a major upgrade packaged as a patch cannot be uninstalled. It is recommended that you thoroughly test the uninstallation of your patch before deploying it to your end users. To learn more about patches that are not uninstallable, see Uninstallable Patches in the Windows Installer Help Library.

Non-Administrator Patches

Project ▪ *This information applies to QuickPatch projects.*

Windows Installer 3.0 and later enables you to create patches that can be installed by non-administrators. Non-administrator patches can be used if all of the following criteria are met:

- The target machine must be running Windows Installer 3.0 or later on the Microsoft Windows XP or later client platform. Server platforms are not supported.
- The application was installed from a removable media such as a CD-ROM or DVD.
- The application was installed in a per-machine context.

Note ▪ *If the ALLUSERS property is overwritten at the command line, non-administrator patches will fail.*

- The base installation must include the certificate that will be used to sign all subsequent patches.
- The base installation must include the `MsiPatchCertificate` table. This table provides the signer certificate that will be used to verify the digital signature of subsequent patches when they are applied by a non-administrator. If necessary, this table can contain multiple certificates, and subsequent patches would need to be able to verify at least one of the certificates. For more information, see Preparing Installations for Non-Administrator Patches.
- The non-administrator patch must contain the `MsiDigitalCertificate` table. This table contains the signing certificates for the signed patches.

If any of the above criteria are not met, end users cannot install the digitally signed patch in a locked-down environment.

A typical scenario in which non-administrator patches are used is the computer game industry. Some computer game users are children who might not have access to areas of the system other than folders in their own user profile and registry keys under HKEY_CURRENT_USER. Their parents would have administrative access to the machines so that they can control what is installed and what their children can access. Parents would install any

and all applications. If patches are available for the installed software, children would be able to download and install non-administrator patches without help from their parents, as long as all of the above criteria have been met.

Creating a QuickPatch Project and Applying QuickPatch Packages

Project ▪ *This information applies to QuickPatch projects.*

A QuickPatch project is recommended for installation authors who want to ship small, single upgrades to their users. QuickPatch authoring provides a simple alternative to creating a patch in the Patch Design view, even though it provides less customization. Fundamentally, both patch creation methods produce the same deliverable types: .msp and .exe files.

To create a QuickPatch project for an existing .msi file or an already existing QuickPatch, use the Create New QuickPatch Wizard. Completing the wizard ensures that all basic requirements for the QuickPatch project are met.

Creating a QuickPatch Project for an Existing QuickPatch

Project ▪ *This information applies to QuickPatch projects.*

You can create a new QuickPatch project based on an existing QuickPatch project. This creates a patch that you can deliver to any end users who need to apply a patch for the original application or specific patched versions of the application.

Caution ▪ *If you open and modify any of the releases listed in the History of the General Information view, your latest project will not work, since intermediate data shared across the releases in the History may have been lost or altered. Therefore, whenever you create a QuickPatch project for an existing QuickPatch project, InstallShield changes the existing QuickPatch project (.ise file) to read-only mode.*

Task ***To create a QuickPatch project for an existing QuickPatch project:***

1. In InstallShield, open the QuickPatch project (.ise file) that you would like to patch.
2. On the **Tools** menu, click **Create QuickPatch**.

Your new QuickPatch project opens in InstallShield.

You can also use the Create New QuickPatch Wizard to create a QuickPatch for an existing QuickPatch.

Specifying Whether to Streamline the QuickPatch Package

Project ▪ *This information applies to QuickPatch projects.*

When you are configuring a QuickPatch project, you have the ability to specify whether you want InstallShield to streamline the creation of your QuickPatch package to build as simple a package as possible. The goal of QuickPatch streamlining is to generate a QuickPatch package that has fewer new subfeatures and custom actions than a non-streamlined QuickPatch package.

For example, if your QuickPatch project includes a new file or registry entry and InstallShield does not use QuickPatch streamlining, InstallShield creates a new subfeature for that file or registry entry. InstallShield also adds one or more prebuilt InstallShield custom actions to work around certain Windows Installer patch requirements. However, if InstallShield does use QuickPatch streamlining, the file or registry entry is added to an existing feature, and no special prebuilt InstallShield custom actions are required.

Note ▪ *InstallShield cannot streamline the creation of a QuickPatch package in the following scenarios:*

- *The QuickPatch package removes an installed file.*
- *The QuickPatch package removes or renames a registry key.*
- *The QuickPatch package targets a non-streamlined QuickPatch image. That is, you cannot use QuickPatch streamlining if you select the check box in the History area of the General Information view for a QuickPatch that did not use QuickPatch streamlining. If you try to build a streamlined QuickPatch that targets one or more non-streamlined QuickPatch images, InstallShield displays a build warning, and it does not use streamlining.*

Also note that you can apply no more than 15 cumulative, non-streamlined QuickPatch packages to a base .msi package or major upgrade package. If you exceed the limit, error 2701 occurs while you are applying the patch. The exact limit is determined by the depth of the package's feature tree and the subfeature that each non-streamlined QuickPatch adds to that tree. Windows Installer limits the depth of the feature tree to 16 levels.

A streamlined QuickPatch package does not contain any new subfeatures, so it does not have this limitation.

Task ***To specify whether to streamline the QuickPatch package:***

1. In the View List under **Define Patch Settings**, click **General Information**.
2. In the **General Information** explorer, click **Build Settings**.
3. Click the **Advanced** tab.
4. In the **Streamline QuickPatch** setting, select the appropriate option:
   - To streamline the QuickPatch package, select **Yes**. This is the default value for new QuickPatch projects.
   - To avoid streamlining the QuickPatch package, select **No**.

Specifying Target Releases for Patching

Project ▪ *This information applies to QuickPatch projects.*

Task ***To specify which releases should be patched by your QuickPatch project:***

1. In the View List under **Define Patch Settings**, click **General Information**.
2. In the **General Information** explorer, click **History**.
3. Select or clear the check boxes next to the intermediate releases—the releases in between the Base QuickPatch Image and the current QuickPatch image—to specify whether or not you want them to be patched by the current project.

Note ▪ *You cannot clear the check box for the Base QuickPatch Image or the current project. If you clear the check box of an intermediate QuickPatch project, your current QuickPatch project cannot be used to upgrade a machine that has that intermediate image as the latest image of your application.*

For example, suppose you distributed version 1.0 as an installation and later released a patch that upgrades the original installation to version 1.1. If you create a version 1.2 QuickPatch and the check box for the 1.1 QuickPatch is not selected in the History, your 1.2 QuickPatch can be used to upgrade the 1.0 release but not the 1.1 release.

Specifying Custom Actions for the QuickPatch to Execute

Project ▪ *This information applies to QuickPatch projects.*

Task ***To specify which custom actions should be run when your QuickPatch is applied:***

1. In the View List under **Define Patch Settings**, click **General Information**.
2. In the **General Information** explorer, click **Custom Action**.
3. Select or clear the check boxes next to the custom actions to specify whether or not you want them to be run during the installation of your QuickPatch.

Adding Files to a QuickPatch

Project ▪ *This information applies to QuickPatch projects.*

Note ▪ *QuickPatch projects do not have support for defining a new target destination for new files that you are adding. Therefore, when you add a new file in a QuickPatch project, the destination location must be a folder that was defined in the original installation.*

Task ***To add a file to your QuickPatch:***

1. In the View List under **Define Patch Settings**, click **Files**.
2. Right-click the **Files To Patch** explorer and then click **Insert New File**. The **Open** dialog box opens.
3. Click the file that you want to add, and then click **Open**. InstallShield adds the file to the **Files To Patch** explorer.
4. Click the new file and then configure its settings.

Specifying Identification Information for a QuickPatch

Project ▪ *This information applies to QuickPatch projects.*

Windows Installer 3.0 and later adds an Add or Remove Programs entry for each QuickPatch package that is applied to a target system, even if the QuickPatch is not uninstallable.

Task ***To specify identification information for a QuickPatch:***

1. In the View List under **Define Patch Settings**, click **General Information**.
2. In the **General Information** explorer, click **Build Settings**.
3. Click the **Identification** tab.
4. Configure each of the settings.

Enabling the Uninstallation of a QuickPatch

Project ▪ *This information applies to QuickPatch projects.*

Windows Installer 3.0 and later supports the uninstallation of QuickPatch packages for small updates and minor upgrades. However, not all QuickPatch packages can be uninstalled. For details about the uninstallation of standard patches and QuickPatch packages, including the associated limitations, see Patch Uninstallation.

Task ***To enable the uninstallation of a QuickPatch package:***

1. In the View List under **Define Patch Settings**, click **General Information**.
2. In the **General Information** explorer, click **Build Settings**.
3. Click the **Common** tab.
4. Select the **Allow Patch to be Uninstalled (Requires Windows Installer 3.0)** check box.

Sequencing QuickPatch Packages

Project ▪ *This information applies to QuickPatch projects.*

Task ***To define a sequence for a QuickPatch package:***

1. In the View List under **Define Patch Settings**, click **General Information**.
2. In the **General Information** explorer, click **Build Settings**.
3. Click the **Advanced** tab.
4. Do one of the following:
   - To use default patch sequencing that InstallShield generates, set the **Create patch sequencing entry** property to **Yes**.
   - To use no patch sequencing, set this property to **No**.

Signing a QuickPatch Package

Project ▪ *This information applies to QuickPatch projects.*

Windows Installer 3.0 and later enables you to create patches that can be installed by non-administrators. Non-administrator QuickPatch packages can be used if strict criteria are met. For example, the base installation that the patch will update must include the certificate that will be used to sign the patch package. To learn about the other criteria that must be met, see Non-Administrator Patches.

Task ***To sign a patch package:***

1. In the View List under **Define Patch Settings**, click **General Information**.
2. In the **General Information** explorer, click **Build Settings**.
3. Click the **Digital Signature** tab.
4. Select the **Sign the Patch Package** check box.

5. Configure the digital signature settings.

Password-Protecting a QuickPatch Package

Project ▪ *This information applies to QuickPatch projects.*

For added security, you can password-protect your QuickPatch package. When you password-protect your QuickPatch package, any end user who wants to apply your QuickPatch must enter a case-sensitive password to launch your update.

Task ***To password-protect a QuickPatch package:***

1. In the View List under **Define Patch Settings**, click **General Information**.
2. In the **General Information** explorer, click **Build Settings**.
3. Click the **Advanced** tab.
4. In the **Password Protect Launcher** setting, select **Yes**.
5. In the **Launcher Password** setting, specify the password that you want your patch to use.

Note ▪ *The password is case-sensitive.*

Specifying Whether to Build an Update.exe Update Launcher for a QuickPatch Package

Project ▪ *This information applies to QuickPatch projects.*

InstallShield lets you specify whether you want InstallShield to create an `Update.exe` update launcher for your QuickPatch package.

An `Update.exe` update launcher is required if you want to automatically update or install the Windows Installer engine on a target system if necessary. For more information on when an `Update.exe` launcher is required, see Patching Considerations.

If you configure InstallShield so that it does not create an `Update.exe` update launcher, it creates an .msp file.

Task ***To specify whether to include an Update.exe update launcher for a QuickPatch:***

1. In the View List under **Define Patch Settings**, click **General Information**.
2. In the **General Information** explorer, click **Build Settings**.
3. Click the **Common** tab.

4. To include `Update.exe`, select the **Create Update.exe** check box.

   To avoid including `Update.exe`, clear the **Create Update.exe** check box.

Tip ▪ *You can also use the Advanced tab in the Build Settings area to specify whether you want to include* `Update.exe`.

Customizing File Properties for the Update Launcher

Project ▪ *This information applies to QuickPatch projects.*

InstallShield lets you use custom information for the version resources of the `Update.exe` update launcher. The information is displayed on the Properties dialog box for the update launcher; this Properties dialog box opens when end users right-click the `Update.exe` file and then click Properties.

Note ▪ *The Properties dialog box is different on different versions of Windows. For example, on Windows 7 systems, the version resource information is displayed on the Details tab of the Properties dialog box. However, on Windows XP systems, the version resource information is displayed on the Version tab of that dialog box.*

Also note that some versions of Windows do not show some settings on the Properties dialog box.

To see screen shots of the Properties dialog box for a `Setup.exe` *file (which looks the same as the Properties dialog box for an* `Update.exe` *file), see Customizing File Properties for the Setup Launcher.*

Task ***To override the default InstallShield values for the Update.exe settings with your own custom values:***

1. In the View List under **Define Patch Settings**, click **General Information**.
2. In the **General Information** explorer, click **Build Settings**.
3. Click the **Advanced** tab, and then find the **Update Launcher Settings** area.
4. In the following settings, enter the values that you want to use for the properties of your `Update.exe` file.
   - Company Name
   - Product Name
   - Product Version
   - Description
   - Copyright

Tip ▪ *The Product Version setting updates the file version and the product version for* `Update.exe`. *Note that the file version always contains four fields. If you specify fewer than four fields, the remaining fields are filled with zeros. For example, if you specify a product version of* **1.1**, *the file version that is used in the version resources of* `Update.exe` *is* **1.1.0.0**.

If you leave the update launcher settings blank, InstallShield uses the default InstallShield values.

Deleting Files from the Files To Patch Explorer

Project ▪ *This information applies to QuickPatch projects.*

Task ***To delete a file from the Files To Patch explorer:***

1. In the View List under **Define Patch Settings**, click **Files**.
2. Right-click the file that you want to delete and then click **Delete**.

Modifying and Removing Installed Files with a QuickPatch

Project ▪ *This information applies to QuickPatch projects.*

Task ***To modify or remove an installed file with your QuickPatch:***

1. In the View List under **Define Patch Settings**, click **Files**.
2. Right-click the **Files To Patch** explorer and then click **Patch Existing File**. The **Select File** dialog box opens.
3. Click the file that you would like to modify or delete. InstallShield adds the file to the **Files To Patch** explorer.
4. Click the file that you just added to the **Files To Patch** explorer and then configure its settings.

Tip ▪ *As an alternative to steps 3 and 4 above, you can drag and drop files or folders from the* ***Original Setup Files*** *explorer to the* ***Files To Patch*** *explorer.*

Adding, Modifying, and Deleting Registry Data with a QuickPatch

Project ▪ *This information applies to QuickPatch projects.*

When you add, modify, or delete registry data with a QuickPatch project, the procedures are basically the same as the ones that you perform for the original installation. The one difference is that before you can add registry data for a QuickPatch project, you must select an existing feature in the View Filter list at the top of the Registry view. All registry data must be associated with a feature that already exists in the original product because you cannot add new features with a QuickPatch.

Tip ▪ *To change any registry settings that you modified for the QuickPatch project, you can right-click the item and then click Revert.*

Patching Assemblies in the Global Assembly Cache

Project ▪ *This information applies to QuickPatch projects.*

With Windows Installer 3.0 or later, the `MsiPatchOldAssemblyFile` and `MsiPatchOldAssemblyName` tables enable a patch package to patch an assembly in the global assembly cache (GAC) without making a run-time request for the original installation source. By default, InstallShield automatically generates entries for these tables when you build a QuickPatch package. (To disable this automatic generation in a QuickPatch project, set the **Generate MsiPatchOldAssembly tables** property to No. In a QuickPatch project, this property is on the Advanced tab of the Build Settings item in the General Information view.)

Note ▪ *To automatically generate entries for the* `MsiPatchOldAssemblyFile` *and* `MsiPatchOldAssemblyName` *tables, InstallShield must have write access to the latest version of the .msi package to which the patch applies. InstallShield modifies this package before creating the patch. InstallShield automatically attempts to make this package writable; if it cannot, a build warning is generated, and the table entries are not created.*

These table entries are applied only if the target system is running Windows Installer 3.0 or later. The patch runs if the system is running Windows Installer 2.0; however, these tables are ignored, and if the patch needs to update a file in the GAC, it makes a request for the original source package. InstallShield generates these table entries even if you include only the Windows Installer 2.0 engine in `Update.exe`*, since a target system with Windows Installer 3.0 or later already installed can use them.*

Applying a QuickPatch

Project ▪ *This information applies to QuickPatch projects.*

A QuickPatch package (.msp) file contains the transforms and instructions necessary for upgrading one or more installed versions of a product.

Before you can apply a QuickPatch package, Windows Installer version 1.1 or later must be present on the system. In addition, the package that you want to update must be installed locally or as an administrative image. A further requirement is that the existing installation package be installed with the same privileges and for the same users as the QuickPatch package. For example, if the product was installed for all users, the update should be installed for all users, as well.

The easiest way to apply a QuickPatch is to double-click the .msp file in Windows Explorer or right-click the file and then click Open. When you apply a patch for a minor upgrade, a PatchWelcome dialog is the first dialog that opens. When you apply a patch for a major upgrade, the full dialog sequence appears, as when you run an installation standalone.

From the command line, you can apply a patch with the MsiExec.exe `/p` option. Type the following statement to apply a patch package located at `X:\Product Updates\Build 36\PatchForV1.msp`:

```
msiexec /p "X:\Product Updates\Build 36\PatchForV1.msp"Update.exe
```

If you selected the Create Update.exe check box for your QuickPatch configuration in the Build Settings area of the General Information view, InstallShield wraps your .msp file in an executable file. `Update.exe` launches your patch package with the following command-line expression:

```
msiexec /p <path to .msp file> REINSTALL=ALL REINSTALLMODE=omus
```

Applying a QuickPatch in Silent Mode

There are two ways you can apply a QuickPatch in silent mode: either launch MsiExec.exe with the `/qn` command-line parameter, or pass `/s` to `Update.exe`.

There is an important consideration to bear in mind when applying a QuickPatch in silent mode. In order to operate correctly, the Windows Installer property `REINSTALL` must be set to **`ALL`** and `REINSTALLMODE` to **`omus`** whenever you apply a QuickPatch. Since `Update.exe` always sets these properties at the command line, you do not have to do anything extra if your QuickPatch package is applied with `Update.exe`.

When a QuickPatch package is applied with a full user interface, one of your installation's default dialogs, PatchWelcome, is displayed. It includes control events to set `REINSTALL` and `REINSTALLMODE` with the correct options. However, since this dialog is not displayed when the end-user interface is suppressed, you must set the properties at the command line, as demonstrated below:

```
msiexec /p <path to .msp file> /qn REINSTALL=ALL REINSTALLMODE=omus
```

Because a QuickPatch does not modify the existing cached .msi database, including the **v** setting for `REINSTALLMODE` is unnecessary.

Notifying End Users about Upgrades Using FlexNet Connect

FlexNet Connect provides a mechanism that enables you to automatically notify your Web-connected end users when patches, updates, and product information for your application are ready for release.

FlexNet Connect Implementation

There are two cycles of tasks that you perform when using FlexNet Connect to automatically inform end users about updates: initial deployment and update deployment. Once you have performed the initial deployment steps for your application, each time you want to distribute an update of that application to your customers, you perform the update deployment cycle of steps.

Update Deployment

1. Use InstallShield to create an update for your application.
2. Register your new product version and product code with the FlexNet Connect Publisher site, a Web-based management portal.
3. Publish your update to the FlexNet Connect Publisher site, and set the **Message Status** to **Test**.
4. Test your update.

5. Publish your update to the FlexNet Connect Publisher site, and set the **Message Status** to **Active**.

FlexNet Connect includes a variety of options that can be purchased together for a complete solution, or separately for a customized solution. To learn more, visit the Revenera Web site.

6

Additional Installation Options

In addition to helping you create an installation package that installs products, InstallShield provides other installation options to help you enhance your final installation package. Read more about additional installation options in this section.

Specifying Target System Requirements

InstallShield enables you to easily set installation requirements for the target system. For example, if your product requires a specific operating system or application in order to run properly, you can indicate that in your installation project. If the target system does not meet the requirement, the installation exists.

Specifying Operating System Requirements for Your Product

InstallShield enables you to specify that your product requires a specific operating system. If the target system does not meet the requirement, the installation exists.

Task ***To specify operating system requirements for target systems:***

1. In the View List under **Define Setup Requirements and Actions**, click **Requirements**.
2. In the center pane, click **System Hardware Requirements**.
3. In the **OS Version** setting, click the ellipsis button (**...**). The **System Hardware Requirements** dialog box opens.
4. Specify the appropriate operating system requirements.

InstallShield updates the value of the OS Version setting.

Tip ▪ *When you specify operating system requirements for your product, you are essentially excluding operating systems that do not support your product.*

For example, if you select only the check box for the latest Windows operating system, InstallShield creates a launch condition to exclude the operating systems that you did not select in the Requirements view. With this type of launch condition, future versions of Windows operating systems are supported automatically because they are not excluded in the launch condition.

Specifying Processor Requirements for Your Product

If your product requires a certain processor class, you can ensure that your product is not installed on systems with incompatible processors. If the target system has a processor that is less than the minimum that you specified, the installation exits.

Task ***To specify processor requirements for target systems:***

1. In the View List under **Define Setup Requirements and Actions**, click **Requirements**.
2. In the center pane, click **System Hardware Requirements**.
3. In the **Processor** setting, select the appropriate processor.

Specifying RAM Requirements for Your Product

If your product requires a certain amount of RAM in order to run properly, you can ensure that your product is not installed on systems with insufficient memory. If the target system does not have the minimum amount required, the installation exits.

Task ***To specify RAM requirements for target systems:***

1. In the View List under **Define Setup Requirements and Actions**, click **Requirements**.
2. In the center pane, click **System Hardware Requirements**.
3. In the **RAM** setting, select the appropriate value.

Specifying Screen Resolution Requirements for Your Product

If your product requires a certain screen resolution in order to run properly, you can ensure that your product will not be installed on systems with lower screen resolutions. If your installation is run on a machine with a lower resolution than what you specify, the installation exits.

Task ***To specify screen resolution requirements for target systems:***

1. In the View List under **Define Setup Requirements and Actions**, click **Requirements**.
2. In the center pane, click **System Hardware Requirements**.
3. In the **Screen Resolution** setting, select the appropriate value.

Specifying Color Depth Requirements

If your product requires a certain color depth in order to run properly, you can ensure that your product will not be installed on systems with lower color depths. If the target system has a color depth that is less than the minimum selected, the installation exits.

Task ***To specify color depth requirements for target systems:***

1. In the View List under **Define Setup Requirements and Actions**, click **Requirements**.
2. In the center pane, click **System Hardware Requirements**.
3. In the **Color Depth** setting, select the appropriate value.

Specifying Software Requirements for Your Product

If your product requires other software in order to run properly, you can ensure that your product will not be installed on systems that do not have that software. If your installation is run on a system that does not have the required software, the installation displays an error message that explains the missing software, and the installation exits.

The System Software Requirements explorer in the Requirements view contains some built-in requirement conditions for software such as Adobe Reader and the .NET Framework. You can add any of these built-in requirements to your project, or you can configure your own requirements.

Task ***To add one of the existing software requirements to your project:***

1. In the View List under **Define Setup Requirements and Actions**, click **Requirements**.
2. In the **System Software Requirements** explorer, locate the software condition that you want to add to your project.

To edit a condition status, click the check box next to the condition until the appropriate icon appears next to it:

Table 6-1 ▪ Icons for the System Software Requirement Conditions

Icon	Description
	The software condition must be true in order for the installation to proceed.
	The software condition must be false in order for the installation to proceed.
	The software condition is not checked and does not affect the installation.

Tip ▪ *To add a new software condition to the existing requirements list, right-click the* ***System Software Requirements*** *explorer and click* ***Create New Condition (System Wizard)****. When you do this, the System Search Wizard opens, enabling you to create a custom requirement. For more information, see System Search Wizard.*

Working with Windows Installer Properties

Windows Installer properties enable you to use variables for common paths and user information throughout your installation. These properties can be used in dialog text, .ini file changes, custom actions, and registry entries. For a list of these properties, see Windows Installer Property Reference.

Whenever these properties are used, they should be surrounded by square brackets. For example, to display the installation directory on a dialog, you may enter the following in the Text and Messages view:

```
This program will be installed to [INSTALLDIR]
```

To the end user, this sentence appears as follows:

```
This program will be installed to C:\Program Files\Your Company\Your Product
```

Property Types

There are four general types of Windows Installer properties:

- Public
- Private
- Restricted public
- Required

Note ▪ *Some of these categories overlap. For example, the* `ProductCode` *property is a required private property.*

Public Properties

Public properties have names that contain only uppercase letters. For example, `INSTALLDIR` is a public property. Public properties can be specified at the command line used to launch the installation or chosen by using an authored user interface.

Note ▪ *Only public properties have their values preserved from an installation's user interface to the point where the installation is changing the target system. If you set the value of a property in a dialog displayed to the end user, use a public property (for example, MY_PUBLIC_PROPERTY) if you want its value written to a file or to the registry.*

Private Properties

Private properties have at least one lowercase letter in their name and cannot be changed from the user interface. For example, ProgramFilesFolder is a private property. End users have no control over the values of private properties, since they cannot be set from the command line.

Restricted Public Properties

Restricted public properties allow network administrators to define public properties that can be changed only by a system administrator. This way, the administrator can change settings quickly without having to worry that other users on the network may tamper with the setup. For more information, see Specifying that a Public Property Should Be a Restricted Public Property.

Required Properties

The Windows Installer service relies on five properties that are required in every Windows Installer installation. By default, these properties are included in every installation that you create using InstallShield.

- ProductCode
- ProductLanguage
- Manufacturer
- ProductVersion
- ProductName

Conditions

Many properties are not set until the installation is launched. These properties are populated with information from the target system. For example, the `VersionNT` property is not set until the installation is launched. This property is set to the version of Windows that the target machine is running if the operating system is Windows 2000 or later.

Properties that are set at run time can be used to create launch conditions for the installation. If you want your product to be installed only on Windows XP, you can use conditional logic to check the end user's system, and install the product if all conditions are met.

Windows Installer Property Reference

Windows Installer properties let you reference certain aspects of your installation through the Windows Installer service. These properties can be used in custom actions, .ini file keyword values, and in dialog text. For information on how to use these properties, see Working with Windows Installer Properties.

The following categories of Windows Installer properties are described in this topic. Click a link to go directly to a category.

- Special Folder and File Properties
- Feature Installation Properties
- Other Configurable Properties
- User-Supplied Information
- Product-Specific Properties
- System Folders Set by the Installer
- Operating System Properties Set by the Installer
- Hardware Properties Set by the Installer
- Status Properties Updated by the Installer
- Date and Time Properties

***Note** ▪ Do not confuse installer properties with path variables, which are surrounded by angle brackets (<>). While they both represent directories, you can use Windows Installer properties at run time, but path variables are used to point to source files only during setup design and at build time.*

Special Folder and File Properties

Special folder properties are those properties that define where files are stored or installed on the target system. File properties refer to specific files.

To use a folder property in your installation, enclose it in square brackets: [].

Table 6-2 ▪ Folder Properties

Property Name	Description
INSTALLDIR	This property contains the default destination folder for the files in your features. For more information, see Setting the Default Product Destination Folder (INSTALLDIR).

Table 6-2 ▪ Folder Properties (cont.)

Property Name	Description
SETUPEXEDIR	The SETUPEXEDIR property contains the path to Setup.exe. For example, if the path to Setup.exe is C:\MySetups\MyApp\Setup.exe, the value of SETUPEXEDIR is C:\MySetups\MyApp. ***Note ▪*** *There are two limitations to using* SETUPEXEDIR*:* ● SETUPEXEDIR *is set by* Setup.exe*. If the end user runs the .msi package directly,* SETUPEXEDIR *is not set. To account for this, you could have a dual implementation in your installation—one that uses* SETUPEXEDIR *and one that uses* SourceDir*. You could test for the existence of* SETUPEXEDIR *and, if it does not exist, you could conditionally use your* SourceDir *implementation.* ● SETUPEXEDIR *might not be set at uninstallation. If the end user triggers the uninstallation by running* Setup.exe*,* SETUPEXEDIR *is set. If they run the uninstallation from Add or Remove Programs,* SETUPEXEDIR *is not set.* Note that in an uncompressed installation, SourceDir and SETUPEXEDIR have the same value.
SETUPEXENAME	This property identifies the name of the setup launcher file that was created when the project was built. The installation updates the value of this property at run time if the setup launcher file was renamed. The following path identifies the full path to this file: [SETUPEXEDIR]\[SETUPEXENAME]
SourceDir	This property stores the root directory where all the source files are located. ***Caution ▪*** *If your release consists of an .msi package that is compressed into a setup launcher and the .msi package is not cached on the local machine,* SourceDir *should be referred to during installation, but not during uninstallation or maintenance. In these circumstances, the setup image runs from a temporary location that is removed following installation.* SETUPEXEDIR is an alternative to the directory identifier SourceDir. A potential problem with using SourceDir is that it points to the location of the running .msi package. In the case of a compressed installation, the .msi package is streamed to a temporary location and run from there. Because of this, the value of SourceDir will be some temporary location on the end user's machine, which might not be the value you want.

Table 6-2 ▪ Folder Properties (cont.)

Property Name	Description
TARGETDIR	The TARGETDIR property is the location where the Windows Installer package is copied (and its data files uncompressed) during an Administrative installation.

Feature Installation Properties

The following section provides information on feature installation properties, with which the end user can specify how features should be installed.

Table 6-3 ▪ Feature Installation Properties

Property Name	Description
ADDDEFAULT	The ADDDEFAULT property stores a list of features, separated by commas, that are to be installed in their default configuration. Users can install all features in their default configuration by setting the value of this property to ALL.
ADDLOCAL	This property stores a list of features, separated by commas, that are to be installed locally. Each feature has a Remote Installation property that determines whether features selected for installation are to be installed locally or run from the source medium.
ADDSOURCE	This property stores a list of features, separated by commas, that are to be run from the source medium. If this property is set to ALL, all the features will run from the source medium.
ADVERTISE	This property stores a list of features, separated by commas, that are to be advertised.
REINSTALL	This property stores a list of features, separated by commas, that are to be reinstalled. If REINSTALL is set to ALL, all of the features that are already installed on the user's system will be reinstalled.
REINSTALLMODE	This property contains a string of letters that specify the reinstall type. These options correspond to the values available for the Msiexec.exe /f command-line parameter. For more information on this property, see REINSTALLMODE Property in the Windows Installer Help Library. REINSTALLMODE is always set in conjunction with REINSTALL.

Table 6-3 ▪ Feature Installation Properties (cont.)

Property Name	Description
ReinstallModeText	This property contains the reinstallation options that will be set when the user selects Repair in the MaintenanceType dialog. The value of ReinstallModeText is passed as an argument to the control event ReinstallMode, which accepts the same options that are available for the `REINSTALLMODE` property. ReinstallModeText is not a predefined Windows Installer property.
REMOVE	This property stores a list of features, separated by commas, that will be removed. If `REMOVE` is set to `ALL`, all features will be removed.
COMPADDLOCAL	This property stores a list of component IDs, separated by commas, that are to be installed locally. The feature for the component that takes up the least amount of disk space will be installed.
COMPADDSOURCE	This property stores a list of component IDs, separated by commas, that are to be run from the source medium. The feature for the component that takes up the least amount of disk space will be installed.
PATCH	When installing a patch, this property contains the full path to the patch package.

Other Configurable Properties

The following section contains information on various other configurable properties.

Table 6-4 ▪ Additional Configurable Properties

Property Name	Description
ACTION	This property specifies which sequence to perform (installation, advertisement, or administration). Possible values are INSTALL, ADVERTISE, and ADMIN. The `ACTION` property is automatically set based on the command line used to launch the installation.

Table 6-4 ▪ Additional Configurable Properties (cont.)

Property Name	Description
ALLUSERS	Specifies whether Windows Installer should attempt to perform a per-machine or per-user installation. If the value of ALLUSERS is set to 1, Windows Installer attempts a per-machine installation. For per-machine installations, configuration information such as shortcuts and registry entries are stored in the All Users profile: ● On Windows Vista and later systems, if User Account Control is enabled and the user does not have administrative privileges, the user must be able to provide administrative credentials in order to install the product. ● On other systems, if the user does not have administrative privileges, the installation displays an error message and exits. ***Project ▪*** *This property is set to 1 by default in all new Express projects. To learn more, see Per-User vs. Per-Machine Installations.* If the value of ALLUSERS is not set or it is an empty string (“”), Windows Installer performs a per-user installation, and the configuration information is stored in the user’s personal profile. If the value of ALLUSERS is set to 2, Windows Installer attempts to perform a per-machine installation on Windows Vista and later systems. On earlier platforms, if the user has administrative privileges, Windows Installer attempts to perform a per-machine installation; otherwise, Windows Installer performs a per-user installation.
ARPAUTHORIZEDCDFPREFIX	This property stores the URL for the update channel of the application.
ARPCOMMENTS	This property contains the comments about this product that are displayed in Add or Remove Programs.
ARPCONTACT	This property contains support contact information, such as an email address or a telephone number.
ARPINSTALLLOCATION	This property stores the fully qualified path to the application’s primary folder. You can set ARPINSTALLLOCATION to the value of INSTALLDIR with a custom action of type “Set a property”.
ARPNOREPAIR	If this property is set to 1, no Repair button will be displayed in the Programs Wizard.

Table 6-4 ▪ Additional Configurable Properties (cont.)

Property Name	Description
ARPREADME	Holds the fully qualified path or the URL for the product's Readme file.
ARPSIZE	This property stores the estimated size, in kilobytes, of the application.
ARPSYSTEMCOMPONENT	Set this property to 1 to keep your program from appearing in Add or Remove Programs.
ARPURLINFOABOUT	This property stores the URL for the application's home page or the publisher's home page.
ARPURLUPDATEINFO	This property stores the URL for update information on your application.
ARPNOMODIFY	Setting this property disables Add or Remove Programs functionality that would, by default, allow the product to be modified.
ARPNOREMOVE	Setting this property disables the Add or Remove Programs functionality that allows the product to be removed.
AVAILABLEFREEREG	This property allows you to set the amount of additional free registry space, in kilobytes, required by your application.
CCP_DRIVE	This property holds the root path on the installation disk for any of the qualifying products for a competitive upgrade.
DISABLEADVTSHORTCUTS	Setting this property disables the creation of advertised shortcuts.
DISABLEMEDIA	This setting prevents the installer from adding media information to the source list.
DISABLEROLLBACK	Set this property to 1 to stop the installer from creating a rollback script that will save copies of changed or deleted files during the installation process.
EXECUTEACTION	This property sets the top-level action initiated by the ExecuteAction action.
EXECUTEMODE	This property sets the mode of execution performed by the installer. The value None means that no changes are made to the system. The default value, Script, means that all changes made to the system are run through a script execution.

Table 6-4 ▪ Additional Configurable Properties (cont.)

Property Name	Description
INSTALLLEVEL	This property holds a value that corresponds to the values of the features. If the level of features to be installed is less than or equal to the `INSTALLLEVEL` property, then a feature is installed. This is primarily used for different setup types, such as Typical or Custom.
LOGACTION	List of action names, separated by semicolons, that will be logged.
MSIINSTALLPERUSER	This property indicates that the Windows Installer should install the package for only the current user: ● If the **ALLUSERS** property is set to 2 and **MSIINSTALLPERUSER** is set to an empty string (""), the Windows Installer performs a per-machine installation. ● If the **ALLUSERS** property is set to 2 and **MSIINSTALLPERUSER** is set to 1, the Windows Installer performs a per-user installation. This property is available with Windows Installer 5 and on Windows 7 or Windows Server 2008 R2. Earlier versions of Windows Installer and Windows ignore this property. For more information, see the following: ● Per-User vs. Per-Machine Installations ● MSIINSTALLPERUSER Property on the MSDN Web site
Privileged	This property will run an installation with elevated privileges if the user is an administrator or if the application is an administrator-assigned application.
PROMPTROLLBACKCOST	This property specifies what will happen if there is insufficient disk space to continue the installation. Depending on the user interface level, the rollback could happen automatically, without any input from the user, or it could ask the user to continue with rollback disabled.
PRIMARYFOLDER	The folder that you specify with this property will be the “primary” folder for the installation. The path to this folder will be used to determine the values for the `PrimaryVolumePath` property, the `PrimaryVolumeSpaceAvailable` property, the `PrimaryVolumeSpaceRequired` property, and the `PrimaryVolumeSpaceRemaining` property.

Table 6-4 ▪ Additional Configurable Properties (cont.)

Property Name	Description
REBOOT	This property allows you to force or suppress a reboot after the installation completes. Possible values are: • **F**—To force a reboot when your installation is complete. • **S**—To suppress any reboot except one caused by the ForceReboot action. • **R**—To suppress any reboot caused by Windows Installer actions.
ROOTDRIVE	In Administration mode this property sets the default drive to the first writable network drive found. In all other modes, this property sets the default drive to the writable local drive with the most disk space available.
SCRIPTFILE	This property defines the location of the script file that contains all operations executed during the installation.
SEQUENCE	This property specifies an .msi database table that lists the order in which the actions in the table will run.
SHORTFILENAMES	In Administration mode, this property may be set to ensure that only short file names are used.
LIMITUI	Setting this property limits the user interface level at basic. This is useful if you do not create a custom user interface to interact with the installer's built-in UI.
DefaultUIFont	This property should be set to one of the predefined styles found in the TextStyle table in order to specify your default font. If this property is not set, the installer will use the system font, which may disrupt your formatting.

User-Supplied Information

The following section contains information about input taken from the end user. Such input can include the end user's name, company, or language.

Table 6-5 ▪ User-Supplied Information

Property Name	Description
AdminProperties	`AdminProperties` holds a list of properties set during an administration installation. These properties can be external (user name) or they can be internal (other properties on this page).

Table 6-5 ▪ User-Supplied Information (cont.)

Property Name	Description
COMPANYNAME	This property stores the organization name for the end user performing the installation. This information is taken from the Customer Information dialog.
ISX_SERIALNUM	This property stores the serial number that the end user enters in the Serial Number field on the Customer Information dialog.
UserLanguageID	This property retains the default language identifier for the end user.
USERNAME	This property stores the name of the end user performing the installation, which is taken from the Customer Information dialog.
ProductLanguage	This property stores the numeric language ID for the product.

Product-Specific Properties

Information on product-specific properties that can be set in the Property table is listed below. Examples of these types of properties include technical support numbers, product name, and serial number.

Table 6-6 ▪ Product-Specific Properties

Property Name	Description
ARPHELPLINK	This property retains the Internet address for technical support. This value is set in the Add/Remove Programs `Support URL` property. You should provide a string table entry to facilitate globalizing your setup.
ARPHELPTELEPHONE	This property retains your technical support telephone numbers. This value is set in the Add/Remove Programs `Support Phone Number` property. You should provide a string table entry to facilitate globalizing your setup.
ProductCode	The ProductCode is the GUID for this particular version of the product. This ID must be different for different language versions and different release versions. This property is set in the **General Information** view.
ProductName	This property stores the name of the product—for example, InstallShield. This property is set in the **General Information** view.

Table 6-6 ▪ Product-Specific Properties (cont.)

Property Name	Description
ProductState	The installer sets this property to the installed state of the product. This property can hold one of four numeric values: **-1**—The product has not been installed or advertised. **1**—The product has been advertised, but not installed. **2**—The product has been installed for another user. **5**—The product has been installed and is available to the current user.
ProductVersion	The `ProductVersion` property stores the major, minor, and build version numbers in the format *AA.BB.CCCC*. This property is set in the General Information view.
Manufacturer	Stores the name of the product manufacturer. This value is set in the Add/Remove Programs `Publisher` property. You should provide a string table entry to facilitate globalizing your setup.
DiskPrompt	This property holds a string which is displayed by a message box prompting for a disk. You should also include empty text for additional information printed on the disk label, as in “Disk 1”.
DiskSerial	The `DiskSerial` property should be set to the internal serial number for this release.
ComponentDownload	This property retains the URL for downloading a product by its string identifier (GUID).
LeftUnit	This property places units to the left of the number. This is necessary for languages that require this structure.
UpgradeCode	This is a GUID used to search for a related set of products that are already installed.
IsAdminPackage	This property is set to 1 if the current installation package was created through an administrative installation. You can use this property to detect post-administrative installations.

System Folders Set by the Installer

The following properties hold the fully qualified path to many of the folders on the end user's system.

Table 6-7 ▪ System Folder Properties

Property Name	Description
AppDataFolder	This property holds the full path to the current user's Application Data folder.
CommonAppDataFolder	This property holds the full path to the All Users Application Data folder.
CommonFilesFolder	The value of this property is the full path to the 32-bit Common Files folder.
DesktopFolder	This property is used to hold the full path to the Desktop folder for the current user. If the setup is being run for All Users, and the ALLUSERS property is set, then the DesktopFolder property should hold the full path to the All Users desktop folder.
FavoritesFolder	The `FavoritesFolder` property retains the full path to the Favorites folder for the current user.
FontsFolder	This property holds the full path to the Fonts folder.
PersonalFolder	This property holds the full path to the current user's Personal folder.
ProgramFilesFolder	This property holds the full path to the current user's Program Files folder.
ProgramMenuFolder	This property is used to hold the full path to the **Program** menu for the current user. If the setup is being run All Users, and the `ALLUSERS` property is set, then the `ProgramMenuFolder` property should hold the full path to the All Users Programs menu.
SendToFolder	This property holds the full path to the current user's SendTo folder.
StartMenuFolder	This property is used to hold the full path to the Start menu folder for the current user. If the setup is being run for All Users, and the `ALLUSERS` property is set, then the `StartMenuFolder` property should hold the full path to the All Users Start Menu folder.
StartupFolder	This property is used to hold the full path to the Startup folder for the current user. If the setup is being run for All Users, and the `ALLUSERS` property is set, then the `StartupFolder` property should hold the full path to the All Users Startup menu.

Table 6-7 ▪ System Folder Properties (cont.)

Property Name	Description
SystemFolder	This property holds the full path to the 32-bit System folder.
TempFolder	This property holds the full path to the Temp folder.
TemplateFolder	This property holds the full path to the current user's Template folder.
WindowsFolder	This property holds the full path to the user's Windows folder.
WindowsVolume	This property is set to the drive where Windows is installed.

Operating System Properties Set by the Installer

The following properties are set by the installer at run time. They refer to environment variables on the target system.

Table 6-8 ▪ Operating System Properties

Property Name	Description
AdminUser	This property is set by the installer at installation and is set if the user has administrative privileges.
ComputerName	This property stores the name of the computer that the installation is running on. It is set by a call to the Windows API **GetComputerName** at the initialization of the installer.
LogonUser	This property stores the name of the user performing the installation. It is set by a call to the Windows API **GetUserName**.
OLEAdvtSupport	This property is set by the installer during initialization if the target system supports install-on-demand through COM.
ServicePackLevel	If an operating system service pack is installed, this property stores the numeric value for that update.
SharedWindows	This property is set when `Shared Windows` is being used on the target system.
ShellAdvtSupport	This property is set by the installer during initialization if the target system supports feature advertisement. This property is automatically set on Windows 98 or later, or on earlier systems if Internet Explorer 4.01 is installed.
SystemLanguageID	This property stores the default language identifier for the target system. The value is defined by the installer by calling GetSystemDefaultLangID at initialization.

Table 6-8 ▪ Operating System Properties (cont.)

Property Name	Description
TerminalServer	This property is set by the installer at initialization if the target system is a server with Windows Terminal Server.
TTCSupport	This property is set by the installer at initialization if the target system supports true type font collections (TTC). The following systems support TTC: JPN - 932, Taiwan - 950, China - 936, Korea - 949, Hong Kong - 950.
Version9X	This property stores the version number of Windows 95 and 98 operating systems as an integer. The following formula is used to determine this integer: (MajorVersion * 100) + MinorVersion. On Windows 95, Version9X is set to 400, on Windows 98 it is set to 410, and on Windows Millennium Edition it is set to 490. Version9X is not set on Windows NT-based systems.
VersionDatabase	This property stores a version number of the database used during the installation.
VersionNT	This property stores the version number of Windows NT-based operating systems as an integer. The following formula is used to determine this integer: (MajorVersion * 100) + MinorVersion. Refer to the Windows Installer Help Library to learn the VersionNT property for a specific operating system.
WindowsBuild	This property stores the build number for the operating system being run.
MsiNTProductType	This property stores the type of NT operating system being run on the target machine. This property requires Windows Installer version 2.0.
MsiNTSuiteBackOffice	This property is set to 1 if Microsoft BackOffice components are installed. In all other cases this property is not set.
MsiNTSuiteDataCenter	This property is set to 1 if Windows 2000 Datacenter Server is installed. In all other cases this property is not set.
MsiNTSuiteEnterprise	This property is set to 1 if Windows 2000 Advanced Server is installed. In all other cases this property is not set.
MsiNTSuiteEnterprise	This property is set to 1 if Windows 2000 Advanced Server is installed. In all other cases this property is not set.
MsiNTSuiteSmallBusiness	This property is set to 1 if Microsoft Small Business Server is installed. In all other cases this property is not set.

Table 6-8 ▪ Operating System Properties (cont.)

Property Name	Description
MsiNTSuiteSmallBusinessRestricted	This property is set to 1 if Microsoft Small Business Server is installed with the restrictive client license. In all other cases this property is not set.
MsiNTSuitePersonal	This property is set to 1 if the operating system is Workstation Personal. In all other cases this property is not set.
MsiNetAssemblySupport	This property is set if the operating system supports .NET Framework assemblies. In all other cases this property is not set. This property requires Windows Installer version 2.0.
MsiWin32AssemblySupport	This property is set if the operating system supports Win32 assemblies. In all other cases this property is not set. This property requires Windows Installer version 2.0.

Hardware Properties Set by the Installer

The following properties are set by the installer at run time and store settings on certain hardware profiles for the end user's system.

Table 6-9 ▪ Hardware Properties

Property Name	Description
Alpha	This property stores the numeric value of the processor level, and it is only defined if the setup is running on an Alpha processor. (This property is supported only with Windows Installer version 1.0.)
BorderSide	This property sets the pixel width of the side window borders.
BorderTop	This property sets the pixel width of the top window border.
CaptionHeight	This property sets the pixel height of the caption area.
ColorBits	This property stores the number of adjacent color bits for each pixel (that is, the color depth of the user's monitor). For example, if the user's monitor is using 256 colors, ColorBits is set to 8.
Intel	This property stores the numeric value of the processor level, and it is defined only if the setup is running on an Intel 32-bit processor.
PhysicalMemory	This property stores the installed amount of physical memory, in megabytes.
ScreenX	This property defines the width of the screen, in pixels.

Table 6-9 ▪ Hardware Properties (cont.)

Property Name	Description
ScreenY	This property defines the height of the screen, in pixels.
TextHeight	This property sets the height of text characters.
VirtualMemory	The amount of available page file space, in megabytes, is stored in this property.

Status Properties Updated by the Installer

The following properties are set by the installer at run time. The values of these properties have to do with the status of the installation.

Table 6-10 ▪ Status Properties

Property Name	Description
AFTERREBOOT	This property is set to 1 by the installer after a reboot triggered by the ForceReboot action.
CostingComplete	This property is set to 1 as soon as costing has begun and is set to 0 when costing is complete.
RollbackDisabled	The installer sets the `RollbackDisabled` property whenever rollback has been disabled. This property is not set by default.
Installed	This property determines if the product is already installed.
OutOfDiskSpace	This property is set to True if any of the drives that are targets for the install do not have enough disk space. Otherwise, it is set to False.
OutOfNoRbDiskSpace	This property is set to True if any disk targeted during an installation does not have enough free disk space and if rollback capability is turned off. It is set to False if all of the target disks have sufficient space.
Preselected	This property determines if features have been preselected, and if so, does not display the selection dialog.
PrimaryVolumePath	The installer sets this property to the path specified in the PRIMARYFOLDER property.
PrimaryVolumeSpaceAvailable	This installer sets this property to a string representing the total number of bytes, in units of 512, available on the volume specified by the `PRIMARYFOLDER` property.

Table 6-10 ▪ Status Properties (cont.)

Property Name	Description
PrimaryVolumeSpaceRequired	This property stores a string representing the total amount of disk space required, in bytes (expressed in units of 512 bytes), by the currently selected features.
PrimaryVolumeSpaceRemaining	The installer sets this property to a string representing the number of remaining bytes available on the system, in units of 512, if all selected features were to be installed.
Resume	This property stores the text string displayed to the user when an installation is resumed from a suspended setup.
UpdateStarted	This property is set once changes to the system have taken place as a result of the installation process.
ReplacedInUseFiles	This property is set if a file that is currently in use is overwritten. Custom actions that check to see if a reboot is required will use this property.
NOUSERNAME	Setting this property to 1 stops the installer from setting the `USERNAME` property. By default, this property is not set and the `USERNAME` property is set from the registry.
NOCOMPANYNAME	Setting this property to 1 stops the installer from setting the `COMPANYNAME` property. By default, this property is not set and the `COMPANYNAME` property is set from the registry.

Date and Time Properties

Table 6-11 ▪ Date and Time Properties

Property Name	Description
Date	This property holds the current date.
Time	This property holds the current time.

Specifying that a Public Property Should Be a Restricted Public Property

Restricted public properties allow network administrators to define public properties that can be changed only by a system administrator or by someone who has elevated privileges. This way, the administrator can change settings quickly without having to worry that unauthorized users on the network may tamper with the installation.

Windows Installer considers a number of public properties to be restricted public properties. For the full list of restricted public properties, see Restricted Public Properties in the Windows Installer Help Library.

To include additional public properties, they must be added to the `SecureCustomProperties` property.

If you perform tasks such as the following ones through InstallShield user interface, InstallShield automatically adds the applicable property to the `SecureCustomProperties` property:

- When you use a public property in a launch condition that you added to your project through the Requirements view, InstallShield adds the public property to the `SecureCustomProperties` property.
- For dialogs that contain a control that is set through a property, InstallShield adds that property to the `SecureCustomProperties` property.

This enables the custom public properties to be set in the User Interface sequence and then passed to the Execute sequence.

Implementing Serial Number Validation at Run Time

As part of your installation, you can require end users to enter a valid serial number. You can do this through a custom action .dll file and the Customer Information run-time dialog.

Task ***To implement serial number validation for your product:***

1. In the View List under **Customize the Setup Appearance**, click **Dialogs**.
2. In the **Dialogs** explorer, click **Customer Information**.
3. Set the value of the **Show Serial Number** field to **Yes**.
4. In the **Serial Number Template** field, specify the format for your product's serial number. For example, type the following:

   ```
   ??????-####-??????????
   ```

   Note the following guidelines for this field:

   - Type a question mark (?) to represent each alphanumeric character.
   - Type a number sign (#) to represent each number.
   - Type a dash (-) between each group of characters. The dash indicates a break in the serial number, where one group of characters ends and another begins.
5. In the **Serial Number Validation DLL** field, enter the location of the .dll file that you use for serial number validation. Alternately, click the browse (**...**) button to locate the .dll file.

   Tip ▪ *Sample code for a serial number validation .dll file is available in the `InstallShield Program Files Folder`\Samples\WindowsInstaller\ValidateSerialNumber directory. For more information, see Using a Custom Action for Serial Number Validation.*
6. In the **Validate Function** field, enter the name of the .dll file function that performs the validation.
7. Set the **Success Return Value** field to a number other than 0.

8. In the **Retry Limit** field, specify the number of times that you want to allow an end user to attempt to enter the serial number.
9. Configure the **Show All Users Option** field as appropriate.

When an end user enters a serial number in the Customer Information dialog, it is passed to your .dll file for verification. If the serial number is successfully verified, the installation can continue. If the serial number is invalid, the installation will end, depending on how many retries you allow in the Retry Limit field.

Note ▪ *InstallShield sequences the serial number .dll file custom action after any user-defined custom actions. Thus, if you add your own custom action and schedule it after the Customer Information dialog, the installation launches this custom action before the serial number .dll file custom action.*

Accessing the Serial Number During and After Installation

The Windows Installer property that stores end user's entry for a serial number is called `ISX_SERIALNUM`. If you select Yes for the Show Serial Number setting for the Customer Information dialog, the serial number field is included on this dialog, and Windows Installer sets the property to the serial number that the end user enters at run time.

During Installation

During installation, you can access the value of `ISX_SERIALNUM` by using code such as in this VBScript sample:

```
' Get the value of Serial Number
Dim sSerialNo
sSerialNo= Session.Property("ISX_SERIALNUM")

' Show it.
MsgBox sSerialNo
```

Important ▪ *If you want to be able to access the value of* `ISX_SERIALNUM` *after the installation, your installation must store the* `ISX_SERIALNUM` *value on the target system somewhere. For example, in the Registry view, you could create a registry key that adds a value that uses* ***[ISX_SERIALNUM]****.*

After Installation

Following installation, you can access the serial number if your installation wrote the value of the `ISX_SERIALNUM` property to the target system. For example, if your installation wrote the value to the registry, you can write code that checks the registry for the serial number.

7

Building InstallShield Project in DevOps

InstallShield supports building the InstallShield projects in DevOps. InstallShield provides the following to build the InstallShield projects:

- Azure DevOps Build Extension
- InstallShield StandAlone Build with Docker

Azure DevOps Build Extension

InstallShield supports the Azure DevOps Build Extension. By using this extension, InstallShield projects can be built in the Azure DevOps pipelines.

Task ***To build the Install the InstallShield project in Azure DevOps pipelines,***

1. Add and install the InstallShield Build Extension to your Azure DevOps organization.
   The **Install InstallShield SAB Task** and **InstallShield Build Task** are available to your Build Pipeline tasks.
2. Configure the tasks.
3. Build the InstallShield projects stored in the repository.

Install InstallShield SAB Task

The Install InstallShield SAB task installs the InstallShield Standalone Build setup on the Microsoft hosted machine, based on the InstallShield version configured in the task, and configures the license server with the license server details provided in the task.

To configure the Install InstallShield SAB Task, see **Task #1: Install InstallShield SAB Task** in the InstallShield Azure DevOps Build Extension KB article.

InstallShield Build Task

The InstallShield Build task checks the InstallShield SAB is installed on the agent or not, if not, it installs the InstallShield SAB and triggers the InstallShield project build.

To configure the InstallShield Build Task, see **Task #2: Install Build Task** in the InstallShield Azure DevOps Build Extension KB article.

InstallShield StandAlone Build with Docker

InstallShield project can also be built via Docker Container. To build InstallShield project via Docker Container, download Docker Image and build the Docker image. For more detail, see InstallShield StandAlone Build with Docker KB article.

8

Integrating InstallShield with External Applications

An important aspect of InstallShield is how it coexists and integrates with other software development tools such as Visual Studio as well as source code control software that complies with the Microsoft Source Control Interface.

Consult the help topics in this section for specific details on the extent of external application support in InstallShield.

Integrating with Microsoft Visual Studio

With InstallShield, you can create your installation projects directly in Microsoft Visual Studio. InstallShield enables you to create, modify, or build your installation from within Microsoft Visual Studio.

Integration Features

InstallShield is fully integrated within the Visual Studio shell. Some of the unique features of the integration effort include:

- All InstallShield navigation is presented within the Solution Explorer.
- Each InstallShield view is presented in a separate window so no scrolling is necessary and side-by-side viewing options are available.
- You can run InstallShield external to Visual Studio.
- Dynamic links to other Visual Studio projects exist, so the new contents of output groups are included in the installation.

Integration Benefits

Some additional benefits of using InstallShield's integrated installation authoring solution are:

- You can create and customize your installation without leaving the Visual Studio user interface, enabling use of familiar navigation and layout options.

- Your installation is automatically updated with your latest source files every time your solution is built, always staying current.
- The installation reflects the Build Configuration for the solution—for example, Debug, Release, automatically included source files from the proper build directory.
- .NET attributes and dependencies can be scanned and included in an installation automatically.

Note ▪ *If you want to create, edit, and build your InstallShield projects directly within Visual Studio, you must use Visual Studio 2005 or later. InstallShield cannot be integrated with Visual Studio 2003 or earlier.*

Visual Studio can be integrated with only one version of InstallShield at a time. The last version of InstallShield that is installed or repaired on a system is the one that is used for Visual Studio integration.

Creating InstallShield Projects in Microsoft Visual Studio

InstallShield is integrated with Microsoft Visual Studio. From within the Visual Studio workspace, you can create InstallShield installations for solutions.

Task ***To create an InstallShield project from within Microsoft Visual Studio:***

1. On the **File** menu, point to **New** and click **Project**. The **New Project** dialog box opens.
2. *Beginning with Visual Studio 2010:* In the **Installed Templates** box, click **InstallShield Express Projects**. Then select the appropriate project type.

   For earlier versions of Visual Studio: In the **Project Types** box, click **InstallShield Express Projects**. Then in the **Templates** box, select the appropriate project type.
3. Configure the settings for the name and project location as needed.
4. Click **OK**.

Opening InstallShield Projects in Microsoft Visual Studio

InstallShield is integrated with Microsoft Visual Studio. From within the Visual Studio workspace, you can open InstallShield installations for solutions.

Task ***To open an InstallShield project from with Microsoft Visual Studio:***

1. On the **File** menu, point to **Open** and then click **Project**. The **Open Project** dialog box opens.
2. Browse to the InstallShield file you want to open.
3. Click **Open**.

Note ▪ *Files created in InstallShield Professional or pre-3.x versions of InstallShield Express cannot be opened.*

Using the VSSolutionFolder Path Variable with Visual Studio Solutions

A predefined path variable called VSSolutionFolder is available in projects to reference a higher-level base directory. This support enables you to have in your InstallShield projects static links to files in sibling projects that are within the Visual Studio solution folder; if you work on the projects on a different machine, the static links that use the VSSolutionFolder path variable can reference the correct paths for the files in sibling projects.

The VSSolutionFolder path variable is defined automatically whenever an InstallShield project is opened from within a Visual Studio solution. It is also defined automatically if you are using MSBuild to build a solution that contains an InstallShield project. However, in other scenarios, when the InstallShield project is opened without the Visual Studio solution, VSSolutionFolder cannot be defined automatically. For example, if you open the InstallShield project in InstallShield directly, without having Visual Studio open, VSSolutionFolder is not defined. Similarly, if you use the command-line tool `IsCmdBld.exe`, or if you use MSBuild with an .isproj file, VSSolutionFolder is not defined. If you are using MSBuild to build a release in InstallShield project, use the PathVariables parameter to set the value of VSSolutionFolder. This parameter is exposed as the ItemGroup InstallShieldPathVariableOverrides when the default targets file is used.

If you include in your InstallShield project a source file whose path includes the VSSolutionFolder path variable and build it in an environment that does not support the VSSolutionFolder path variable, build errors such as the following ones may occur:

- -6103: Could not find file <VSSolutionFolder>\MyFile.exe
- -6271: File <VSSolutionFolder>\MyFile.exe not found. An error occurred building the MsiFileHash table record for this file. Verify that the file exists in the specified location.

Adding References to Visual Studio Solutions

Use the Files view to add Visual Studio references to your installation project.

Task ***To add a reference:***

1. In the **Solution Explorer** under **Specify Application Data**, double-click **Files**.
2. The Visual Studio Solution item in the **Source computer's folders** pane contains subitems for all projects contained in the current solution. Click a project to display the output groups for the project. The output groups are displayed in the **Source computer's files** pane.
3. To add a reference to an output to your installation project, drag and drop the output to the target folder in the **Destination computer's folders** pane.

Adding InstallShield Toolbars or Commands to the Visual Studio Toolbar

You can add InstallShield toolbars and individual toolbar command buttons to the Microsoft Visual Studio workspace.

Adding InstallShield Toolbars

Task ***To add an InstallShield toolbar:***

1. Right-click anywhere in the toolbar to display the toolbar options.
2. Select a toolbar to add it to the top of the Visual Studio workspace.

Adding InstallShield Toolbar Command Buttons

Task ***To add an individual command button to the toolbar:***

1. Right-click anywhere in the toolbar to display the toolbar options.
2. Select **Customize** from the bottom of the list. The **Customize** dialog box opens.
3. Click the **Commands** tab.
4. In the **Categories** list, select a category to display the commands available within that particular category.
5. In the **Commands** list, click a command button and drag it to the toolbar.

Building Releases in Microsoft Visual Studio

Building a release of an InstallShield project in Visual Studio is different than building a release in InstallShield. When you are building in Visual Studio, you have the option to either build your entire solution, including the installation project, or build only the installation project.

Before you can build a release, the release must be associated with a solution configuration. The Configuration Manager in Visual Studio is where you associate a release with a solution configuration.

Task ***To build a release from within Visual Studio:***

1. Use the Configuration Manager in Visual Studio to map a release to the appropriate solution configuration.
   a. On the **Build** menu, click **Configuration Manager**.

      As an alternative: On the **Standard** toolbar, in the **Solution Configurations** list, select **Configuration Manager**.

   b. In the **Active solution configuration** list, select a configuration.

   c. In the **Project contexts** box, map the project configuration to the appropriate release in the **Configuration** column.

      Note ▪ *If the Build check box is cleared and you build the solution, the deselected project configuration is not built.*

   d. Click the **Close** button.

2. On the **Build** menu, select the appropriate command:

Table 8-1 ▪ Build Menu Commands

Command	Description
Build Solution	Build the entire solution, according to what is mapped in the Configuration Manager for each project that is included in the solution.
Build *Installation Project Name*	Build only the installation project, according to the release that is specified in the Configuration column of the Configuration Manager. This option is available if your installation project is selected in the Solution Explorer.

Tip ▪ *To build the entire solution, you can also press CTRL+SHIFT+B.*

Adding .NET Assemblies to a Project

InstallShield enables you to add a .NET assembly to your installation project by adding the .NET assembly file to a feature.

Task ***To add a .NET assembly to your project:***

1. In the **Solution Explorer** under **Specify Application Data**, double-click **Files**.
2. Add the .NET assembly to a feature.
3. Right-click the file and then click **Properties**. The **Properties** dialog box opens.
4. Click the **COM & .NET Settings** tab.
5. In the **Scan at Build** list, select **Dependencies and Properties**.

At build time, InstallShield scans the .NET assembly and adds its dependencies and properties as needed.

Adding Project Output from Web Services or a Web Application

InstallShield provides enhanced Web services support. If you add a project output (any project output) from a Web Service or Web Application project, InstallShield will prompt you to add the project as a Web Service. If you choose No, the project output that you have selected is added normally. If you select Yes, InstallShield performs the following:

1. InstallShield creates a Destination Folder called IISROOTFOLDER.
2. InstallShield deploys the following Visual Studio Project Outputs into the IISROOTFOLDER:

   [Content Files] goes to the [IISROOTFOLDER]{VSIPProjectName}

   [Primary Output] goes to the [IISROOTFOLDER]{VSIPProjectName}\bin"

3. InstallShield creates an IISVirtualDirectory with a target of [IISROOTFOLDER]{VSIPProjectName}.

Adding .NET Framework Support to an Installation Project

If your application installation requires that .NET Framework support on the target system, you can add that support to your installation project. For more information, see Adding .NET Framework Redistributables to Projects.

Integrating with Microsoft Visual Studio Team Foundation Server

Microsoft Visual Studio Team Foundation Server (TFS) is a set of tools and technologies that enable a team to collaborate and coordinate the tasks for developing a product. InstallShield has support for integrating with Team Foundation Server. Some highlights of this integration are:

- **Source control**—Use the Source Control Explorer to integrate your InstallShield project with Team Foundation version control and manage changes to your InstallShield projects and your Visual Studio solutions.
- **Automated builds**—Use Team Foundation Build to compile, test, and deploy your InstallShield projects and your Visual Studio solutions on a regular basis. Your installation is automatically updated with your latest source files every time your solution is built, always staying current.
- **Project management**—Track work items such as bugs, tasks, and project documentation for your InstallShield projects and your Visual Studio solutions. The project status is available to your entire team from within Team System Web Access, and from within Team Explorer.

Integration Requirements

To integrate InstallShield with Team Foundation Server, install InstallShield on each machine that you want to be able to create, update, or build InstallShield projects. Thus, InstallShield should be installed on each machine on which you want to create and update InstallShield projects. It should also be installed on a machine that is designated as a build agent for InstallShield projects that are stored in Team Foundation Server. For InstallShield licensing details, refer to the InstallShield End-User License Agreement (EULA).

In order for a build agent to build some types of projects and solutions, you may need to install additional software on the build machine. For example, to build a C++ project, which requires the C++ compiler and possibly other dependencies, install Visual Studio on the build machine.

If you are using multiple build agents to build your Team Foundation Server projects, you may want to assign a particular build tag to any agents that are on machines that have InstallShield; you could also apply that special build tag to each build definition that is created for an InstallShield project. That way, only build machines that have InstallShield installed would be used to build InstallShield installations. For more information about creating build tags and assigning them to agents and build definitions, see the Visual Studio Team Foundation Server documentation.

If you are queuing a build on a 64-bit build machine, ensure that you have configured the build definition for your InstallShield project so that the 32-bit version of MSBuild is used to load the `InstallShield.Tasks.dll` file (which is a 32-bit file); otherwise, you will encounter a build error informing you that the `InstallShield.Tasks.dll` file

could not be loaded. To select the 32-bit version of MSBuild, click the Process tab of your build definition in Team Explorer. Then, under the Advanced node, find the MSBuild Platform setting, and select x86. Note that if you are using a 32-bit build machine, you can select either Auto or x86 for the MSBuild Platform setting.

If you install Team Explorer on the same machine that has InstallShield and Visual Studio, you can use Team Explorer from within your InstallShield projects that are open in Visual Studio. This enables you to perform tasks such as the following:

- Use Source Control Explorer when you are working on your InstallShield projects.
- Configure builds for your InstallShield projects and Visual Studio solutions.
- Queue new builds.

Note that when you queue a build for a solution that includes an InstallShield project, the installation that is built is copied to an Install subfolder within the drop folder. When the InstallShield build detects that it is running under Team Foundation Build, it copies the installation to the final output location for the solution (OutDir)—namely, the binaries directory, which in turn is copied to the drop folder at the end of the Team Foundation Build process.

Adding InstallShield Projects to Team Explorer

If you have Team Explorer installed on the same machine that has InstallShield and Visual Studio, you can add your InstallShield projects (.ise and .isproj files) to the Team Foundation Server through Team Explorer.

If you want to add an InstallShield project to Source Control Explorer when you create the project from within Visual Studio, select the **Add to source control** check box on the New Project dialog box. To add an existing InstallShield project to Source Control Explorer, add it using the same method that you use for adding other files.

9

Reference

Reference information for InstallShield is organized into the following sections:

Table 9-1 ▪ Reference Selections

Section	Description
Menu, Toolbar, and Window Reference	Describes the various components of the InstallShield user interface, including menus, toolbars, and windows.
Dialog Box Reference	Contains reference information on each of the dialog boxes that are displayed in InstallShield.
Wizard Reference	Provides details about each of the wizards that are available in InstallShield.
View Reference	Describes each of the views that are displayed in InstallShield.
Errors and Warnings	Provides information about error codes and warnings that might occur when you create, build, or run your installation. This section also includes reference information about errors and warnings that may occur when you migrate a project from an earlier version of an InstallShield product to the latest version.
InstallShield Custom Action Reference	Explains each of the custom actions that are available in InstallShield.
Command-Line Tools	Introduces tools that you can use from the command line to perform tasks such as building a release and running an installation.
End-User Dialogs	Serves as a reference for all end-user dialogs available through the Dialogs view in InstallShield.

Menu, Toolbar, and Window Reference

This section describes the various components of the InstallShield user interface, including menus, toolbars, and windows.

Menus

The menus in InstallShield are located on the menu bar, which is at the top of the InstallShield user interface. Each menu contains a list of commands. Some of these commands have icons next to them so that you can quickly associate the command with the icon.

Each of the menus in InstallShield is described in this section:

- File
- Edit
- View
- Go
- Project
- Build
- Tools
- Help

File Menu

The following table lists the File menu commands, as well as associated keyboard shortcuts and icons.

Table 9-1 ▪ File Menu Commands

Command	Shortcut	Icon	Description
New	Ctrl + N		Creates a new installation project.
Open	Ctrl + O		Opens an existing installation project.
Close			Closes the current project.
Save	Ctrl + S		Saves the current project.
Save As			Enables you to save the current project file with a new name and location.

Table 9-1 ▪ File Menu Commands (cont.)

Command	Shortcut	Icon	Description
1, 2, 3, 4			Opens one of the most recently accessed projects.
Exit			Closes the current project and closes InstallShield.

Edit Menu

The following table lists the Edit menu commands, as well as associated keyboard shortcuts.

Table 9-2 ▪ Edit Menu Commands

Command	Shortcut	Description
Undo	Ctrl + Z	Undoes the last action that was performed.
Cut	Ctrl + X	Cuts the currently selected text to the clipboard.
Copy	Ctrl + C	Copies the currently selected text to the clipboard.
Paste	Ctrl + V	Pastes the clipboard contents.
Find	Ctrl + F	Opens the Find dialog box, which enables you to search for folders and files that you have added to your installation project in the Files view. For more information, see Finding Files and Folders in Your Project.

View Menu

The following table lists the View menu commands, as well as associated keyboard shortcuts and icons.

Table 9-3 ▪ View Menu Commands

Command	Shortcut	Icon	Description
Output Window			Toggles the display of the Output window.
View List			Toggles the View list.
View Bar	F4		Toggles the View bar.
Header Bar			Toggles the header bar.
Toolbar			Toggles the toolbar.
Status Bar			Toggles the status bar.

Table 9-3 ▪ View Menu Commands (cont.)

Command	Shortcut	Icon	Description
Project Assistant			Displays the Project Assistant.

Go Menu

The following table lists the Go menu commands, as well as associated keyboard shortcuts and icons. Some of the view-related commands are not available from the Go menu, depending on which project type you have open in InstallShield.

Table 9-4 ▪ Go Menu Commands

Command	Shortcut	Icon	Description
Previous View	Alt + Up Arrow		Takes you to the view directly above the current view as shown in the View List.
Next View	Alt + Down Arrow		Takes you to the view directly below the current view as shown in the View List.
Back	Alt + Left Arrow		Takes you to the view that you last visited in the history of your view selections. You can use this multiple times, as long as there are multiple entries in your view history.
Forward	Alt + Right Arrow		Takes you to the next view in the history of your view selections. You can continue until you reach the view you were at when you first clicked Back.
Start Page			Takes you to the Start Page.
Help			Takes you to the Help view.
Project Assistant			Displays the Project Assistant.
Organize Your Setup			Enables you to display the General Information, Features, Setup Types, Update Notifications, or Upgrade Paths view.
Specify Application Data			Enables you to display the Files, Files and Features, Redistributables, or Dependencies view.

Table 9-4 ▪ Go Menu Commands (cont.)

Command	Shortcut	Icon	Description
Configure the Target System			Enables you to display the Shortcuts/Folders, Registry, ODBC Resources, INI File Changes, File Extensions, Environment Variables, Internet Information Services, or Component Services view.
Customize the Setup Appearance			Enables you to display the Dialogs, Billboards, or Text and Messages view.
Define Setup Requirements and Actions			Enables you to display the Requirements, Custom Actions, or Setup Files view.
Prepare for Release			Enables you to display the Releases view.

Project Menu

The following table lists the Project menu commands.

Table 9-5 ▪ Project Menu Commands

Command	Description
Visual Studio Deployment Project Import Wizard	Launches the wizard that lets you import a Visual Studio setup or merge module project into an InstallShield project.
Perform Static Scan	Launches the Static Scanning Wizard
Perform Dynamic Scan	Launches the Dynamic Scanning Wizard
Import String Entries	Launches the Import String Table Wizard.
Export String Entries	Launches the Export String Table Wizard
Settings	Launches the Msi Log File Settings dialog box.

Build Menu

The following table lists the Build menu commands, as well as associated keyboard shortcuts and icons.

Table 9-6 ▪ Build Menu Commands

Command	Shortcut	Icon	Description
Build	F7		Builds your release with default settings, or—if you have already built your release—rebuilds your release with your most recently saved settings.
Quick Build	Shift + F7		Rebuilds only the .msi file part of your installation to allow you to see your changes take effect more quickly than a complete rebuild.
Stop Build	Ctrl + Break		Cancels the current build process.
Test	Ctrl + T		Allows you to run through the user interface portion of your installation without making any changes to your system. All custom actions are executed.
Run	Ctrl + F5		Allows you to run your completed installation without leaving the IDE.
Uninstall			Uninstalls the most recently run release.
Web Deployment Wizard	Ctrl + W		Launches the Web Deployment Wizard to create an installation that can be launched from a Web page.

Tools Menu

The following table lists the Tools menu commands, as well as associated icons.

Table 9-7 ▪ Tools Menu Commands

Command	Icon	Description
Open Release Folder		Launches Windows Explorer, and opens the release folder.
Create QuickPatch		This prompts to save and close any existing project, and then it will launch the QuickPatch Wizard.
Check for Updates		Checks for service packs and other updates to InstallShield. If updates are available, you can choose to download and install.

Table 9-7 ▪ Tools Menu Commands (cont.)

Command	Icon	Description
Redistributable Downloader		Launches the Redistributable Downloader Wizard to quickly download third-party redistributables, merge modules, prerequisites, and other files to your local machine.

Help Menu

The following table lists the Help menu commands, as well as associated icons.

Table 9-8 ▪ Help Menu Commands

Command	Icon	Description
Contents		Displays the Contents tab of the InstallShield Help Library.
Index		Displays the Index tab of the InstallShield Help Library.
Search		Displays the Search tab of the InstallShield Help Library.
Support Central		Displays Support Central on the Web.
InstallShield Community		Displays the Community on the Web.
Release Notes		Displays InstallShield release notes.
Revenera on the Web		Connects to the Revenera Web site.
Help View		Displays the Help view.
About InstallShield Express		Displays the About InstallShield dialog box, where you can find version information and register InstallShield.

Toolbars

The InstallShield user interface offers the Standard toolbar, which gives you quick access to frequently used menu commands. You can create your own custom toolbars or customize the Standard toolbar to fit your needs. Toolbars can be resized and repositioned, and docked and undocked.

Standard Toolbar

The following table describes all of the buttons on the Standard toolbar.

Table 9-9 ▪ Standard Toolbar Buttons

Button	Name	Description
	New Project	Launches the New Project dialog box, which enables you to select a project type and begin a new project.
	Open	Launches the Open dialog box, which enables you to open an existing installation project.
	Save	Saves the current project file.
	Viewbar	Hides or shows the viewbar, which appears on the left side of the interface.
	View List	Hides or shows the View List, which shows all the views available in the InstallShield interface. ***Note ▪*** *You can also press F4 to hide or show the View List.*
	Previous View	Displays the view directly above the current view, as shown in the View List.
	Next View	Displays the view directly below the current view, as shown in the View List.
	Back	Displays the view that you last visited in the history of your view selections. You can click this button multiple times, if there are multiple entries in your view history.
	Forward	Displays the next view in the history of your view selections. You can continue clicking this button until you reach the view in which you first clicked the Back button.
	Build	Builds your release with default settings, or—if you have already built your release—rebuilds your release with your most recently saved settings.
	Stop Build	Cancels the current build process.

Table 9-9 ▪ Standard Toolbar Buttons (cont.)

Button	Name	Description
	Run	Runs your completed installation project. ***Note ▪*** *If you selected the* ***Uninstall before installing*** *check box, InstallShield uninstalls your product before rerunning the installation. This check box is available on the Preferences tab of the Options dialog box.*
	Uninstall	Uninstalls the most recently run release.
	Test	Runs through the user interface portion of your installation project without making any changes to your system. All custom actions are executed.
	Web Deployment Wizard	Launches the Web Deployment Wizard to create an installation that can be launched from a Web page.
	Open Release Folder	Launches Windows Explorer, open in the DISK1 folder of the current release. If there is no release or if a release has not been built, Windows Explorer opens to your default project location.
	Help View	Displays the IDE Help view where you can find answers to many of your questions regarding InstallShield.

Output Window

The Output window opens across the bottom of InstallShield when you build your project. It also provides information about your project during project conversion. The following tabs appear in the Output window:

Table 9-10 ▪ Output Window Tabs

Tab	Description
Output	Stores distribution output information and displays build output; a link to the output file saved as a text file will be active.
Tasks	Provides descriptions of any errors and warnings that occur when you build your project; each error or warning code displayed will link to an article in the Knowledge Base.

Dialog Box Reference

Each of the dialog boxes available in the user interface of InstallShield is described in this section:

- .NET 1.1/2.0 Core Language Dialog Box
- .NET 1.1/2.0 Language Packs Dialog Box
- Add MIME Type Dialog Box
- Application Extension Mapping Dialog Box
- Application Mappings Dialog Box
- Browse for a Destination File Dialog Box
- Browse for Directory/Set INSTALLDIR Dialog Box
- Browse for File Dialog Box
- Browse for Shortcut Target Dialog Box
- Browse for Tile Target Dialog Box
- Certificate Selection Dialog Box
- Condition Builder Dialog Box
- Content Source Path Dialog Box
- Custom Errors Dialog Box
- Dependencies Dialog Box
- Dialog Images Dialog Box
- Digitally Sign Setup Dialog Box
- Edit Registry Data Dialog Box
- Error Mapping Properties Dialog Box
- File Details Dialog Box
- File Properties Dialog Box
- File Removal Properties Dialog Box
- Folder Properties Dialog Box
- InstallShield Prerequisites Properties Dialog Box
- Logging Options for Windows Installer 4.0 and Later Dialog Box
- Merge Module Configurable Values Dialog Box
- Merge Module Properties Dialog Box
- MIME Types Dialog Box
- MSI Value Dialog Box
- Multi-Line String Value Dialog Box

- New Project Dialog Box
- Options Dialog Box
- Outputs Dialog Box
- Permissions Dialog Boxes for Files and Directories
- Permissions Dialog Boxes for Registry Keys
- Select Icon Dialog Box
- Set INSTALLDIR Dialog Box/Set DATABASEDIR Dialog Box
- Settings Dialog Box
- System Hardware Requirements Dialog Box
- Update Merge Module Search Path Dialog Box
- Upgrade Express Project Name Dialog Box

.NET 1.1/2.0 Core Language Dialog Box

Use the .NET 1.1/2.0 Core Language dialog box to select the .NET core language that you want to distribute. This is the language that is used while the .NET 1.1 core redistributable is installed.

If the 2.0 version of the .NET Framework is selected in the .NET Framework Version setting, the language options are all selected and disabled since they are all included with this version of the redistributable.

If you do not have a particular language installed on your system, the check box for that language is disabled in this dialog box.

***Tip** ▪ To download one or more language versions of the .NET Framework redistributables to your system, click the Download More Languages button; doing so launches the Redistributable Downloader Wizard, which enables you to download one or more redistributables to your system.*

.NET 1.1/2.0 Language Packs Dialog Box

Use the .NET 1.1/2.0 Language Packs dialog box to select the languages that correspond with the .NET language packs that you want to be installed on target systems. This dialog box opens when you click the ellipsis button (...) in the .NET 1.1/2.0 Language Packs setting on the .NET/J# tab for a release in the Releases view.

If you do not have a particular language pack installed on your system, the check box for that language is disabled in this dialog box.

***Tip** ▪ To download one or more language packs to your system, click the Download More Languages button; doing so launches the Redistributable Downloader Wizard, which enables you to download language packs to your system.*

Add MIME Type Dialog Box

Use the Add MIME Type dialog box to add or modify the mapping between a file extension and the program or interpreter that processes those files. This dialog box opens when you click Add or Edit on the MIME Types dialog box.

Table 9-1 ▪ Add MIME Type Dialog Box Settings

Setting	Description
File Name Extension	Type the file name extension (for example, **.abc**). This is a static file name extension. To use a wild-card application mapping for an executable file, enter an asterisk (*).
MIME Type	Type the MIME type (for example, **application/octet-stream**).

Application Extension Mapping Dialog Box

Use the Application Extension Mapping dialog box to add or modify the mapping between a file name extension and the program or interpreter that processes those files. This dialog box is launched when you click Add on the Application Mappings dialog box.

Table 9-2 ▪ Application Extension Mapping Dialog Box Settings

Setting	Description
Extension	Type the file name extension that is associated with your application (for example, *.abc*). To use a wild-card application mapping for an executable file, enter an asterisk (*).
Executable	Type the path, or click Browse to launch the Select Executable dialog box. This enables you to specify the executable in your project to which you want to map. Type the name of the executable file (.exe or .dll) or use the Browse button to search for the file. The executable file must be located on your Web server's local hard drive.
Verbs	The Verbs section enables you to specify which HTTP verbs should be passed to the application. • **All verbs**—Select this option to include all verbs. This option passes all requests to an application. • **Limit to**—Select this option to specify the HTTP verbs that should be passed to an application. Separate verbs with a comma.

Table 9-2 ▪ Application Extension Mapping Dialog Box Settings (cont.)

Setting	Description
Script Engine (IIS 6 and earlier only)	If you want the application to run in a directory without Execute permissions, select this check box. This setting is intended primarily for script-based applications, such as ASP and IDC, that are mapped to an interpreter. For a script-mapped application to run, either the Scripts Only or Scripts and Executables option must be selected for the Execute Permissions property. To allow only script-mapped applications to run, select the Scripts Only option. To allow both script-mapped applications and executable files (.exe and .dll) to run, select Scripts and Executables. This setting applies to IIS 6 and earlier. IIS 7 ignores this setting.
Check that file exists (IIS 6 and earlier only)	To instruct the Web server to verify the existence of the requested script file and to specify that the requesting user should have access permission for that script file, select this check box. If the script does not exist or the end user does not have permission, the appropriate warning message is returned to the browser and the script engine is not invoked. This option can be useful for scripts mapped to non-CGI executable files like the Perl interpreter that do not send a CGI response if the script is not accessible. ***Note ▪*** *Because the script is opened twice—once by the server and once by the script engine—there is some performance cost when this check box is selected.* This setting applies to IIS 6 and earlier. IIS 7 ignores this setting.

Application Mappings Dialog Box

The Application Mappings dialog box lets you add, edit, and delete a mapping between a file name extension and the application that processes those files.

The Application Mappings dialog box is available from within the Internet Information Services view. To open this dialog box, click a Web site, application, or virtual directory in the explorer. Then click the ellipsis button (...) in the Application Mappings setting.

Note ▪ *If an asterisk (*) appears in the Verbs column, all verbs will be used for the specified extension.*

Table 9-3 ▪ Application Mappings Dialog Box Settings

Setting	Description
Add	To add an entry for mapping a file name extension and the program or interpreter that processes those files, click this button. This opens the Application Extension Mapping dialog box.

Table 9-3 ▪ Application Mappings Dialog Box Settings (cont.)

Setting	Description
Edit	To edit an existing application mapping, select the mapping and click this button.
Remove	To delete an existing application mapping, select the extension and click this button.

Browse for a Destination File Dialog Box

When you include a custom action with your setup, and if you select "Installed with the product" as the Source Location value, this dialog box appears when you select the File Name property. Select the feature that includes the file you want to use in your custom action, and then locate the file and click Open.

Browse for Directory/Set INSTALLDIR Dialog Box

Use this dialog box to browse to a directory, create a new directory, rename a directory, or delete a directory.

Dialog Box Settings

Destination Directories

This field lists all of the currently available destination directories. You can select, create, rename, or delete directories in this field.

Selecting a Directory

Task ***To select a directory:***

1. Click a directory to select it.
2. Click **OK**.

Creating a New Directory

Task ***To create a new directory:***

1. Select a directory or the **Destination Computer** and press INSERT. InstallShield creates a directory beneath the selected folder or beneath the Destination Computer.
2. Type the directory name.
3. Provide a directory identifier, if necessary.

Renaming a Directory

Task ***To rename a directory:***

1. Select a directory or the **Destination Computer** and press F2.
2. Type the new directory name. Note that you cannot rename predefined directories.
3. Rename the directory identifier, if necessary, to be consistent with the directory's new name.

Deleting a Directory

Task ***To delete a directory:***

Select a directory and press DELETE. Note that you cannot delete predefined directories.

When you delete a directory, any subdirectories beneath the selected directory are also deleted.

Directory Identifier

You can use the Directory Identifier box to provide a user-friendly name for a directory.

Note ▪ *The directory identifier must be a valid MSI identifier. For features, the directory identifier must contain all capital letters.*

Browse for File Dialog Box

The Browse for File dialog box is displayed when you click the ellipsis button (...) in the Include Patterns and Files setting or the Exclude Patterns and Files setting on the Signing tab for a release in the Releases view. The Browse for File dialog box lets you specify which static files in your project should or should not be signed. It also lets you use an asterisk (*) as a wild-card character. This is especially helpful if you include dynamically linked files in your project and you want to sign all files that match a certain pattern.

Table 9-4 ▪ Settings on the Browse for File Dialog Box

Setting	Description
Select files to sign	This box is displayed if you click the ellipsis button (...) in the Include Patterns and Files setting. This box lists all of the statically included files in your project that match the file types that are selected in the **Show files of type** list. It also lists some default file patterns, such as *.dll. Select the check boxes for the files and file patterns that correspond with the types of files in your project that you want InstallShield to sign at build time.

Table 9-4 ▪ Settings on the Browse for File Dialog Box (cont.)

Setting	Description
Select files to skip signing	This box is displayed if you click the ellipsis button (...) in the Exclude Patterns and Files setting. This box lists all of the statically included files in your project that match the file types that are selected in the **Show files of type** list. It also lists some default file patterns, such as *.dll. If you want to prevent certain files or file patterns from being signed by InstallShield at build time, select the check boxes for the appropriate files and file patterns.
Show files of type	Use this list to filter the types of files that are displayed in the **Select files to sign** box or the **Select files to skip signing** box.

Windows Logo ▪ *All executable files (including .exe, .dll, .ocx, .sys, .cpl, .drv, and .scr files) in an installation must be digitally signed for the Windows logo program.*

When you click OK on this dialog box, InstallShield adds new Include settings under the Include Patterns and Files setting or new Exclude settings under the Exclude Patterns and Files setting.

Note that the files and file patterns that should not be signed override any files and file patterns that should be signed. For example, if you specify *.exe in an Include setting and in an Exclude setting, InstallShield does not sign any .exe files.

Browse for Shortcut Target Dialog Box

The Browse for Shortcut Target dialog box enables you to specify a target for a shortcut.

Table 9-5 ▪ Shortcut Target Dialog Box Options

Option	Description
Look In	This box lists all of the currently available destination directories.
File List	This list shows the files that are located in the directory that is identified in the Look In box.
File Name	If you want the target of the shortcut to be a specific file in the selected destination directory, type the file name in this box.
Files of	This list lets you filter the types of files that are displayed in the file list.

Browse for Tile Target Dialog Box

InstallShield supports configuring the appearance of a desktop app's tile on the Start screen. Use the Browse for Tile Target dialog box navigate to and select the .exe file for the app whose tile appearance you want to configure.

Table 9-6 ▪ Browse for Tile Target Dialog Box Settings

Setting	Description
Look in	Browse to the directory that contains the .exe file of the app whose tile appearance you want to configure. Selecting a folder from this list displays the contents of the folder in the primary pane. In the primary pane, click an .exe file to select it.
File Name	After you have selected an .exe file in the primary pane, its file name is displayed here.

Certificate Selection Dialog Box

When you are configuring digital signature information for a release in your project, use the Certificate Selection dialog box to specify which certificate you want to use to sign your files. InstallShield lets you choose between the following options:

- You can specify the .pfx certificate file or certificate file (EV exported .cer file) on your machine that you want to use for signing.
- You can reference a certificate store that contains the certificate that you want to use for signing.

Accessing the Certificate Selection Dialog Box

Instructions on how to access the Certificate Selection dialog box depend on whether you are specifying certificate information for a release or a QuickPatch package.

Task ***To access the Certificate Selection dialog box for a release:***

1. In the View List under **Prepare for Release**, click **Releases**.
2. In the **Releases** explorer, select the release that you want to configure.
3. In the **Digital Certificate Information** setting, click the ellipsis button (**...**).

Task ***To access the Certificate Selection dialog box in a QuickPatch project:***

1. In the View List under **Patch Settings**, click **General Information**.
2. In the **General Information** explorer, click **Build Settings**.
3. Click the **Digital Signature** tab.
4. Next to the **Digital Certificate Information** setting, click the **Browse** button.

Certificate Selection Dialog Box Settings

Table 9-7 ▪ Certificate Selection Dialog Box Settings

Setting	Description
Use a file (.pfx)	To use a .pfx file or certificate file (EV exported .cer file) to digitally sign your release at build time, select this option. Then specify the location of your .pfx or certificate file (EV exported .cer file). You can type the path to the file or use the ellipsis button (...) to browse to the file location.
Use a certificate store	To reference a certificate store that contains the certificate that you want to use to digitally sign your release at build time, select this option and then enter values in the subsettings under this option.
Certificate Store Name	Select the name of the certificate store that contains the certificate that you want to use. Available options are: ● Personal ● Trusted Root Certification Authorities ● Enterprise Trust ● Intermediate Certification Authorities This setting is available if you select the **Use a certificate store** option.
Certificate Store Location	Select the location of the certificate store that contains the certificate that you want to use. Available options are: ● User ● Machine This setting is available if you select the **Use a certificate store** option.
Certificate Subject	Enter the subject of the certificate that you want to use, or select from the list of certificates that are available on your machine. This setting is available if you select the **Use a certificate store** option.
View Details	The details of the certificate like the general information of the certificate, security details and certification path is listed on the **View Details** option.
Signature Digest	Choose the signature digest hashing algorithm (or choose to let InstallShield specify it automatically based on the certificate hash). Available options are: ● Based on certificate hash ● SHA-1 ● SHA-256 ● Dual (SHA-1 & SHA-256)

Condition Builder Dialog Box

The Condition Builder dialog box enables you to create operating system and software conditions for features and custom actions in your project. In addition, this dialog box enables you to create feature conditions for custom actions.

Task ***To launch the Condition Builder dialog box, do one of the following:***

Click the ellipsis button (**...**) in the **Condition** setting for either a feature or a custom action.

Note ▪ *You cannot create conditions for the Always Install feature.*

Dialog Box Settings

Operating System Tab

On the Operating System tab, you can require that the target system run one of the specified operating systems in order to install a feature or run a custom action. If the target system is not running one of the specified operating systems, the feature is not installed or the custom action is not launched.

Software Tab

On the Software tab, you can create software requirement conditions. If the software specified in this tab is not found on the target system, the associated feature is not installed or the custom action is not launched.

Setting Software Requirements

You can set a software requirement condition based on whether the software exists on the target system. To run a custom action or install a feature when the software *is* found on the target system, set the check box to the green checked state ☑. To set the check box to this state, click it until the green check mark appears in the box.

To run a custom action or install a feature only if the software is *not* installed on the target system, set the check box in front of the software to the red “X” (☒). To set the check box to this state, click it until the red “X” appears in the box.

An empty check box (☐) indicates that the condition is not based upon the software in any way. To void a software condition, click the check box until it is empty.

Feature Tab

Note ▪ *The Feature tab is displayed only if you are creating conditions for a custom action.*

On the Features tab, you can select the feature requirements for the custom action’s condition. The custom action will only run if the selected features are set to be installed.

Content Source Path Dialog Box

The Content Source Path dialog box opens when you click the UNC button in the Content Source Path (Local or UNC) setting for a Web site in the Internet Information Services view. This dialog box contains the following setting.

Table 9-8 ▪ Content Source Path Dialog Box Setting

Setting	Description
Universal Naming Convention content source path	Specify the UNC path for the IIS Web site's files. For example: **\\server\share**

Custom Errors Dialog Box

The Custom Errors dialog box lists all of the HTTP errors that can be customized for a Web site, application, or virtual directory. A custom error can be a URL or a pointer to a file on the server.

Task ***To configure error messages:***

1. In the **Custom Errors** dialog box, select one or more errors.
2. Click **Edit**. The **Error Mapping Properties** dialog box opens.
3. Select a message type and specify a file or URL as appropriate.
4. Click **OK**.

You can click the Set to Default button to restore default settings for the selected error.

Dependencies Dialog Box

The Dependencies dialog box displays a list of dependencies when you right-click an item in the **Destination computer's files** pane of the Files view and click **Dependencies from scan at build**.

The dialog box provides results for assembly DLLs. If you launch the dialog box from Microsoft Visual Studio, the dialog box shows results for project outputs. If launched from InstallShield outside of Visual Studio, scan at build dependencies is disabled for project outputs.

Note ▪ *This is enabled only for portable executable files (for example, EXE, DLL, or OCX) and when the .NET Scan at Build property is set to Dependencies and Properties (via the COM and .NET Settings tab in the File Properties dialog box). If the dependency or one of its dependencies cannot be located, a red icon is displayed.*

Dialog Box Settings

Dependency

This section lists all of the dependencies with a check box next to each. To exclude a dependency from the build, deselect the check box next to the dependency. To close the dialog box, click OK.

Note ▪ *Any new dependencies detected at build time—for files added after closing the Dependencies dialog box—are added to the build.*

Dialog Images Dialog Box

Use the Dialog Images dialog box to add an image (.bmp, .gif, .jpg, .jpeg, or .ibd) that you want to display on your installation's dialogs.

Full Screen Image

Browse to a graphic file that will serve as the full-screen background for your exterior dialogs. Exterior dialogs are dialogs that appear at the beginning or end of the installation, including InstallWelcome and SetupCompleteSuccess (the final dialog upon successful installation). The full-screen image should measure 499 by 312 pixels.

Banner Image

Browse to a graphic file that will run across the top of interior dialogs. Interior dialogs, which appear between the first and last installation dialogs, include the License Agreement and Custom Setup dialogs. The banner image should be 499 by 58 pixels.

Digitally Sign Setup Dialog Box

The Digitally Sign Setup dialog box is displayed when you click Digitally Sign Setup on the Build Installation page of the Project Assistant. This dialog box enables you to assure your end users that the code within your application has not been modified or corrupted since publication.

Table 9-9 ▪ Digitally Sign Setup Dialog Box Settings

Setting	Description
Digitally sign setup	To digitally sign your installation, select this check box The other settings on this dialog box are enabled when you select this check box.
Certificate URL	Type a fully qualified URL—for example, **http://www.mydomain.com**. This URL is used in your digital signature to link to a site that you would like end users to visit to learn more about your product, organization, or company.

Table 9-9 ▪ Digitally Sign Setup Dialog Box Settings (cont.)

Setting	Description
Digital certificate information	To specify the digital certificate that you want to use to sign your release, click the ellipsis button (...) next to this setting. The Certificate Selection dialog box opens, enabling you to specify either the location of the .pfx file, certificate file (EV exported .cer file), or information about the certificate store that contains the certificate. InstallShield provides an option to encrypt and store an EV token password in the project file using the public key certificate (.cer) file. The .cer file is generally created by exporting a public key from the EV Authentication Client tools associated with a USB eToken provider (for example, SafeNet Authentication Client). InstallShield displays additional options to configure the Extended Validation (EV) certificate properties if the .cer file is specified. For more details, see ● Configuring Extended Validation (EV) Certificate information in InstallShield ● Certificate Selection Dialog Box After specifying the .pfx file or choosing the certificate from test store, the below will be displayed: ● **Certificate Thumbprint**—This read-only setting displays the certificate thumbprint. ● **Issued By**—This read-only setting displays the certificate issuer information. ● **Expiration Date**—This read-only setting displays the certificate's expiration date. After specifying the certificate file (EV exported .cer file), the below settings will be displayed: ● **Cryptographic Provider**—This setting allows you to specify the cryptographic service provider (CSP). ● **Container Name**—This setting allows you to specify the private key container name associated to the cryptographic service provider (CSP). ● **Token Password**—This setting allows you to specify the EV token password which is encrypted and stored in a project file. ***Note ▪*** *Note the following informations while utilizing the settings that appear after specifying the certificate file (EV exported .cer file):* ● *Both the Cryptographic Service Provider (CSP) name and Container name can be obtained from the Private Key Certificate properties of a user certificate in the EV Vendor Authentication Client tool.* ● *An EV certificate vendor determines an EV token password's expiration period and number of invalid password attempts before it is locked. Therefore, selecting this setting requires changing your password in specific intervals.* ● *If an EV token password is locked, unlocking/resetting the token password requires an administrator password.*

Table 9-9 ▪ Digitally Sign Setup Dialog Box Settings (cont.)

Setting	Description
Password	If the .pfx file that you are using has a password, enter it. InstallShield encrypts the password and stores it in your project file (.ise). At build time, InstallShield uses the password to sign files with a .pfx file. If your certificate is protected by a password but you do not enter it in this setting, signing with a .pfx file fails. Note that if you configure your project to use a certificate that was imported with password protection into a store, Windows prompts for the password at build time when InstallShield is attempting to sign your project's files. The strong key protection that Windows uses does not permit InstallShield to provide the password to the cryptographic provider.

Tip ▪ *The Signing tab in the Releases view lets you specify which portions of your installation should be digitally signed at build time. InstallShield enables you to sign any and all of the following files in a release, depending on what type of project you are using:*

- *Windows Installer package (.msi file) for Express projects*
- `Setup.exe` *file for Express projects*
- *Any files in your release, including your application files*

To learn more, see Digitally Signing a Release and Its Files at Build Time.

Windows Logo ▪ *All executable files (including .exe, .dll, .ocx, .sys, .cpl, .drv, and .scr files) in an installation must be digitally signed for the Windows logo program.*

Edit Registry Data Dialog Box

This dialog box enables you to edit the registry data within your installation project. To launch this dialog box, right-click on a value in the Registry view and click Modify.

Table 9-10 ▪ Edit Registry Data Dialog Box Settings

Setting	Description
Value Name	The value name is read-only in this dialog box. To change the name of a value, exit this dialog box, select the value you want to rename, and press F2.
Value Data	Enter the data for this registry value as you want it to appear on the target machine.

Error Mapping Properties Dialog Box

The Error Mapping Properties dialog box displays an IIS virtual directory error code and its default properties. Use the Message Type list to set the message to a URL or a pointer to a file on the server. When you set the error message to a URL, you must specify the full URL. When you set the error message to a file pointer, you can browse for the file that you want to use as the error message or you can type the full path of the file to be used. This can include files that are not in your project but are pre-existing on the system.

File Details Dialog Box

Access this dialog box by clicking the Details button in the System Search Wizard. This button becomes active after you define a search method and specify the file you are locating. In this dialog box, you have the ability to enhance a search by specifying the following details:

Table 9-11 ▪ File Details Dialog Box Settings

Setting	Description
Minimum Version	The search is successful if the file exists on the target system and the version is higher than the version entered. ***Note ▪*** *The MinVersion property of the signature table will only find a file that is "GREATER THAN" that version. Therefore, if you want to find 1.00.11, the value of your MinVersion property should be 1.00.10. The Windows Installer documentation states this differently.*
Maximum Version	The search is successful if the file exists on the target system and the version is equal to or lower than what is entered.
Minimum Date	Select the check box to search by a minimum date. The search is successful if the file exists on the target system and the date is equal to or later than the date entered.
Maximum Date	Select the check box to search by a maximum date. The search is successful if the file exists on the target system and the date is earlier or equal to the date entered.
Minimum Size	The search is successful if the file exists on the target system and is equal to or larger than the size indicated (in bytes).
Maximum Size	The search is successful if the file exists on the target system and is smaller than or equal to the size indicated (in bytes).
Languages	Click the Browse (...) button to display the Languages dialog box. You can select multiple languages as a condition of your search. The search is successful if at least one of the languages listed is a match.

Note ▪ *The information you provide in the edit fields is optional. You can leave any field blank.*

File Properties Dialog Box

The Properties dialog box for a file determines various attributes that are set for a file when it is installed on a target system.

Note that the settings that are configured on the Properties dialog box cannot be set for dynamically linked files. For more information, see Limitations of Dynamic File Linking.

Task ***To open the Properties dialog box:***

1. In the View List under **Specify Application Data**, click **Files**.
2. Right-click a file and click **Properties**.

The following tabs are available on this dialog box:

- General
- COM & .NET Settings
- Advanced

General Tab

The General tab of the Properties dialog box lets you override various properties for a file when it is installed on the target system.

Note that file properties cannot be set for dynamically linked files. For more information, see Limitations of Dynamic File Linking.

Table 9-12 ▪ Settings for the General Tab on the File Properties Dialog Box

Setting	Description
Location	This read-only setting displays the directory in which this file will be installed on a target system, based on the destination of the component that contains the file.
Font Title	If you are installing a font, you can specify the font title in this box using the format **FontTitle (FontType)**—for example, **Roman (All res)**. InstallShield provides the name of the font if it is registered on your system. You should not specify a font title for .ttf or .ttc fonts because Windows Installer reads the embedded font name and registers it for you. Thus, for .ttf or .ttc files, InstallShield sets the value of this setting to **[Title read from file]**. For more information, see ICE07 in the Windows Installer Help Library.

Table 9-12 ▪ Settings for the General Tab on the File Properties Dialog Box (cont.)

Setting	Description
Override system attributes	To install this file using the same system properties that are currently set for this file on the development system, clear this check box. To override one or more of the file's properties, select the **Override system attributes** check box and then select one or more of the following check boxes. ● **Read-only**—If you want the file to be read-only when Windows Installer installs it, select this check box. ● **Hidden**—If you want the file to be hidden when Windows Installer installs it, select this check box. ● **Use File Hash**—This option is applicable for unversioned files only. Windows Installer can use file hashing to detect and avoid unnecessary copying of unversioned files. If you want Windows Installer to compare the hash of the file in your installation with the hash of the corresponding file on the target system when deciding whether to upgrade an existing file, select this check box. ● **System**—If you want Windows Installer to install the file as a system file, select this check box. ● **Vital**—To indicate that this file is vital to the operation of its component, select this check box. A vital file's component is not installed if the file cannot be installed for any reason. If a vital file cannot be installed, the end user sees an error message with Retry and Cancel buttons, instead of the usual Abort, Retry, and Ignore buttons (which enable the end user to complete the installation successfully without installing that file).
Override system size	The Size setting shows the file's size when the **Override system size** setting is cleared. If you want Windows Installer to ignore the actual size of the selected file when it is calculating the required disk space requirements for your installation, and instead consider the size to be a value that you specify, select this check box and enter the appropriate value (in bytes) in the Size setting.

Table 9-12 ▪ Settings for the General Tab on the File Properties Dialog Box (cont.)

Setting	Description
Override system version	If the file is a versioned file, the Version setting shows the file's version when the **Override system version** setting is cleared. If you want Windows Installer to ignore the actual version number of the selected file, and instead consider the version to be a number that you specify, select this check box and enter the appropriate version number in the Version setting. At run time, if this file has the same name and target location as one on the target system, Windows Installer uses the version number that you specify when it is determining whether to update the target system's file with the version in your current installation, or to leave the file as is. For example, if the file in your project is version 2.0.0.0, and you enter an override version of 3.0.0.0, Windows Installer may replace the file on the target system if it is version 3.0.0.1 or later, but not if it is earlier than 3.0.0.0. For more details about how Windows Installer determines whether to overwrite an existing file, see Overwriting Files on the Target Machine. The maximum version number for a file is 65535.65535.65535.65535.
Override system language	If you want Windows Installer to ignore the language of the selected file and instead consider the language to be one that you specify, select this check box and type the decimal value for the language identifier code in the Language box. At run time, if this file has the same name and target location as one on the target system, Windows Installer considers the language of the file in your installation to be the one that you specified when it is determining whether to update the file with the version in your current installation, or to leave the file as is. For more details about how Windows Installer determines whether to overwrite an existing file, see Overwriting Files on the Target Machine. Font files should not be authored with a language ID because fonts do not have an embedded language ID resource. Leave this entry blank for font files.
Permissions	To set permissions for the file, click this button.

COM & .NET Settings Tab

The COM & .NET Settings tab enables you to set COM and .NET attributes for files in your installation. This tab is available only when InstallShield determines that the file is a portable executable. You can indicate which files you want treated as portable executables on the File Extensions tab of the Options dialog box.

Note that COM and .NET settings cannot be configured for dynamically linked files. For more information, see Limitations of Dynamic File Linking.

Table 9-13 ▪ Settings for the COM & .NET Settings Tab on the File Properties Dialog Box

Setting	Description
Registration Type	This property enables you to indicate how you want your file to be registered. Depending on the file type, you can select from the following options: • **None**—Select this option if you do not want the selected file to be registered on the target machine. This is the default setting for all files. • **Extract COM Information**—Select this option if you want InstallShield to extract all COM registration data from your file and register it on the system during setup. This is the recommended way to register COM objects. • **Self-Registration**—If your file supports self-registration, you can choose this option. Note that self-registration is not as reliable as having Windows Installer register and unregister the file with extracted COM information.
Scan at Build	If you want to scan for .NET dependencies or properties at build-time, select one of the following options: • **None**—Choose this if you do not want to scan for .NET dependencies or properties. • **Attributes Only**—Scan only for .NET attributes. • **Dependencies and Properties**—Scan for both .NET dependencies and properties. Add missing dependencies and properties to the installation project.
Application File	This property is used when your project is scanned at build time. The scanner uses this property along with other information to determine the value of the File Application property for the assembly.
Installer Class	Select this option to ensure that at installation time, the assembly's Install, Commit, Rollback, and Uninstall methods will be called at the appropriate time.
COM Interop	Select this option to enable the .NET COM interop for the assembly. At installation, registry entries are created on the target system that allow COM objects to call your assembly.

Advanced Tab

On the Advanced tab of the File Properties dialog box, you can specify how a file will be registered and which operating systems you want the file installed on.

Note that these settings cannot be configured for dynamically linked files. For more information, see Limitations of Dynamic File Linking.

Table 9-14 ▪ Settings for the Advanced Tab on the File Properties Dialog Box

Setting	Description
Target operating system	There may be cases where you need to install a different version of a file depending on the operating system of the target system. With the **Target operating system** setting, you can specify for which operating systems your file is intended. For more information, see Specifying Operating System Requirements for Your Product. To install your file under any supported platform, select the **All operating systems** check box. This option is selected by default. To select the specific operating systems your file targets, clear this box.
Installation/Uninstallation Properties	In the Installation/Uninstallation Properties area, you can indicate how you want the installer to handle this file with regard to installation and uninstallation. Valid options are: ● **Permanent**—Select this option if you want this file to permanently remain on the target system. The installer does not remove this file during an uninstallation. ● **Shared**—Select the Shared check box to instruct the installer to reference count, or "refcount" this file. When a file is marked as shared, Windows Installer creates a refcount if one does not exist or increments it if it does. While Windows Installer maintains a separate tally of all shared files, the standard refcount is stored under the following registry key: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\SharedDLLs This count is decremented when the file is uninstalled.

Table 9-14 ▪ Settings for the Advanced Tab on the File Properties Dialog Box (cont.)

Setting	Description
File overwrite properties	In the **File overwrite properties** area, you can indicate how you want the installer to handle this file if it already exists on the target system. **Windows Installer Versioning Rules (Recommended)**—If you select this option, Windows Installer versioning rules are used to determine whether a file that already exists on the target system should be replaced. Windows Installer enforces the following rules: • **Versioned files**—In all cases, the file with the highest version is maintained, even if the file already on the target machine has a higher version than the one being installed. Additionally, a file of any version is maintained over unversioned files. • **File language**—All other things being equal, the file that is the same language as the installation is maintained over different language versions of the file. The only exception to this rule applies to multiple language files. Files with multiple languages are maintained over single language versions of a file. • **Date**—If the modified date of a file already present on the target machine is later than the creation date of that file, the file is not overwritten. This rule protects user preference files from being wiped out during an upgrade or reinstallation. **Never Overwrite**—If you select this option, the file—if it exists on the target system—is never overwritten, regardless of the file version. **Always Overwrite**—If you select this option, the file—if it exists on the target system—is always overwritten, regardless of the file version.

File Removal Properties Dialog Box

The file removal Properties dialog box indicates information about a selected file or folder that you want to be removed from target systems. This file and folder removal capability is useful for scenarios such as removing application-created files that your installation does not otherwise track.

Task ***To access the file removal Properties dialog box:***

1. In the View List under **Specify Application Data**, click **Files**.
2. In the **Destination computer's folders** pane, click the folder that contains the file removal item that you want to configure.
3. In the **Designation computer's files** pane, right-click the file removal item that you want to configure, and then click **Properties**.

   InstallShield uses an icon that has a red X through it for identifying files and folders that are configured to be removed.

To learn how to add file and folder removal items to your project, see Removing Files and Folders from Target Systems.

Table 9-15 ▪ Settings in the Properties Dialog Box

Setting	Description
Location	This setting indicates the location of the selected item that is configured to be removed.
Remove the folder if it is empty	If the selected removal item is a folder that you want to be removed from target systems when it is empty, select this option.
Remove files from the folder	If the selected removal item is a file in the folder that is identified in the Location setting and that you want to be removed from target systems, select this option.
File Name	This setting is available if you select the **Remove files from the folder** option. Enter the name of the file that you want to be removed.
Removal Scheduling	Specify when you want the selected file or folder to be removed from target systems. Removals occur when the file or folder's associated feature is processed in one of the following ways: • **Feature installation**—The selected item is removed if its feature is being installed and if—for folder removal items—the folder is empty. • **Feature uninstallation**—The selected item is removed if its feature is being uninstalled and if—for folder removal items—the folder is empty. • **Feature installation and uninstallation**—The selected item is removed if its feature is being installed or uninstalled and if—for folder removal items—the folder is empty.

Folder Properties Dialog Box

This dialog box provides information about the type and location of folders defined in InstallShield, as well as options that determine how files are dynamically acquired when building a setup.

There are two tabs available from this dialog box:

- General
- File Linking

General Tab

The General panel of the Folder Properties dialog box displays information such as the number of files you have placed in the selected directory as well as the overall disk space required by those files. None of the information on this panel can be edited. To display the Folder properties panel, right-click on a destination folder in the Files view and select Properties.

File Linking Tab

The File Linking Tab of the Folder Properties dialog box enables you to specify dynamic folders. This dialog box can be accessed by right-clicking on a destination folder in the Files view and clicking Dynamic File Linking.

Table 9-16 ▪ Settings for the File Linking Tab of the Folder Properties Dialog Box

Setting	Description
Source folder	Enter the full path to the folder that you want to have dynamically linked, or click the Browse button to navigate to it.
Include subfolders	To dynamically link the files in each subfolder, select this check box. For information on how InstallShield creates components for the dynamically linked files in subfolders, see Determining the Appropriate Component Creation Method for Dynamically Linked Files.
Self-register all files	To self-register every file in the dynamic link, select this check box.

Table 9-16 ▪ Settings for the File Linking Tab of the Folder Properties Dialog Box (cont.)

Setting	Description
Create using best practice methods	To specify that InstallShield should adhere to best practices when creating the components for dynamically linked files, select this check box. When best practices for component creation are followed, InstallShield performs the following tasks at build time for all of the files that meet the include and exclude filter criteria: ● InstallShield creates a separate component for each portable executable (PE) file in the dynamically linked folder. Each PE file is the key file of its component. ● InstallShield adds all non-PE files at the root level of the dynamic link to the component that contains the link. ● If the dynamic link includes a subfolder, InstallShield creates a new component for all of the non-PE files in that subfolder. If the dynamic link includes more than one subfolder, InstallShield creates a separate component for all of the non-PE files in each subfolder. To specify that InstallShield should not follow best practices when creating the components for dynamically linked files, clear this check box. For this component creation method, InstallShield performs the following tasks at build time for all of the files that meet the include and exclude filter criteria: ● InstallShield creates one component for all of the files that are in the root-level dynamically linked folder, regardless of the file types. ● If the dynamic link includes one or more subfolders, InstallShield creates a separate component for all of the files in each subfolder, regardless of the file types. The first dynamically linked file in a subfolder's component is the key file of that component. This check box is selected by default for all new dynamic links. ***Tip ▪*** *A component is the smallest installable part of a product. The Express edition of InstallShield creates components for you automatically. For more information, see Installation Fundamentals.* For more information, see Determining the Appropriate Component Creation Method for Dynamically Linked Files.
Include all files	To include the entire contents of your linked directory in your installation, select this option.

Table 9-16 ▪ Settings for the File Linking Tab of the Folder Properties Dialog Box (cont.)

Setting	Description
Include/exclude files based on the following wild-card patterns	To include or exclude file types, select this option. Enter the file extension in the include or exclude field, preceded by an asterisk (*). Separate multiple entries with a comma. For example, if all of your image files are in one folder along with sound files and you want to dynamically link only the image files, you could specify that you would like to include only .bmp and .ico files in the dynamically linked folder. To do so, you would use an asterisk (*) in your include pattern, as in the following example: `*.bmp, *.ico` To include or exclude a specific file, you would enter the full file name in the include or exclude pattern box.

Note ▪ *For information on dynamic file linking limitations, see Dynamic File Linking.*

InstallShield Prerequisites Properties Dialog Box

The InstallShield Prerequisites Properties dialog box opens when you right-click a selected InstallShield prerequisite in the Redistributables view and then click Properties. This dialog box enables you to specify a location for the selected InstallShield prerequisite. To learn more, see Specifying the Run-Time Location for InstallShield Prerequisites at the Release Level.

Build Location

Select the appropriate option from this list. You have three options:

- **Download From The Web**—To download the InstallShield prerequisite files included in your project (if necessary) from the URL specified in the InstallShield prerequisite (.prq) file for each prerequisite, select this option.
- **Extract From Setup.exe**—To compress the InstallShield prerequisite files into `Setup.exe`, to be extracted at run time, if necessary, select this option.
- **Copy From Source Media**—To store the InstallShield prerequisite files at the root directory of the source media, select this option.

Tip ▪ *If you select the Extract From Setup.exe option or the Copy From Source Media option and then build a release that includes an InstallShield prerequisite that is not available on your computer, one or more build errors are generated for every file that the prerequisite requires. To avoid these build errors, either download the InstallShield prerequisite from the Internet to your computer or remove it from your project before building the release.*

Logging Options for Windows Installer 4.0 and Later Dialog Box

InstallShield displays the Windows Installer 4 Logging Options dialog box when you click the ellipsis button (...) for the Create MSI Logs setting in the General Information view. This dialog box enables you to specify on a project-wide basis—without having to use the command line or configure log parameters through the registry—whether Windows Installer should log your installation. You can also use this dialog box to customize the types of messages that are logged.

Table 9-17 ▪ Options for the Logging Options for Windows Installer 4.0 and Later Dialog Box

Option	Description
No	Installations are not logged. This is the default value.
Yes (MsiLogging set to default value of voicewarmupx)	InstallShield populates the `MsiLogging` property with the default value of **voicewarmupx**. If the installation is run on a target system that has Windows Installer 4.0, the following occurs: ● The installer creates a log file according to the default logging mode of voicewarmupx. ● The installer populates the `MsiLogFileLocation` property with the log file's path. ● A **Show the Windows Installer log** check box is added to the SetupCompleteSuccess, SetupCompleteError, and SetupInterrupted dialogs. If the end user selects that check box and then clicks Finish, the log file is opened in a text file viewer or editor. Earlier versions of Windows Installer ignore this setting. The **Show the Windows Installer log** check box is not visible in run-time dialogs that are displayed on systems running earlier versions of Windows Installer.
Custom MsiLogging value	InstallShield populates the `MsiLogging` property with the value that you specify in the box. If the installation is run on a target system that has Windows Installer 4.0, the following occurs: ● The installer creates a log file according to the custom value that you specified in the box. ● The installer populates the `MsiLogFileLocation` property with the log file's path. ● A **Show the Windows Installer log** check box is added to the SetupCompleteSuccess, SetupCompleteError, and SetupInterrupted dialogs. If the end user selects that check box and then clicks Finish, the log file is opened in a text file viewer or editor. Earlier versions of Windows Installer ignore this setting. The **Show the Windows Installer log** check box is not visible in run-time dialogs that are displayed on systems running earlier versions of Windows Installer.

Important ▪ *The* `MsiLogFileLocation` *property is read-only; it cannot be used to set or change the log file location.*

Merge Module Configurable Values Dialog Box

A configurable redistributable is a merge module or an object that has at least one row in the `ModuleConfiguration` table that is referenced by at least one row in the `ModuleSubstitution` table. This enables you to change a value in the redistributable.

Displaying the Merge Modules Configurable Values Dialog Box

The Merge Module Configurable Values dialog box is displayed when you select a merge module or object in the Redistributables view that allows configuration. If the redistributable selected is an object, the Merge Module Configurable Values dialog box is displayed when the object wizard finishes.

You can also right-click on a configurable merge module or object and click **Configure merge module**.

Dialog Box Settings

The dialog box contains a grid in which you can modify the configurable values. The left column contains the name of the configurable value. The right column contains the value options. You can select a new value from the drop-down menu. Refer to the redistributable vendor's documentation for information about the values.

After you specify the values that you want, click OK to save the new values. When you build your project, these values are used to build the .msi package.

Restore Defaults

Click this button to restore the redistributable's default settings. All of the configurable values in the redistributable are returned to their defaults.

Merge Module Properties Dialog Box

The Merge Module Properties dialog box is displayed when you right-click a selected merge module or object in the Redistributables view and click Properties.

Dialog Box Settings

GUID

Displays the merge module's unique GUID.

Author

Displays the merge module's author.

Version

Displays the merge module version.

Destination

Specifies the destination of the merge module's files. It is recommended you use the default setting **(Use merge module's default destination)**.

MIME Types Dialog Box

The MIME Types dialog box lets you add, edit, and delete mappings between file name extensions and the corresponding content types that are served as static files by the Web server to a browser or mail client.

The MIME Types dialog box is available from within the Internet Information Services view. To open this dialog box, click a Web site, application, or virtual directory in the explorer. Then click the ellipsis button (...) in the MIME Types setting.

Table 9-18 ▪ MIME Types Dialog Box Settings

Setting	Description
Add	To add an entry for mapping a file name extension and the corresponding content type that is served as a static file by the Web server to a browser or mail client, click this button. This opens the Add MIME Types dialog box.
Edit	To edit an existing MIME type, select the MIME type and click this button.
Delete	To delete an existing MIME type, select the MIME type and click this button.

MSI Value Dialog Box

The MSI Value dialog box enables you to create or modify a primary key for an entry in the resulting Windows Installer package's Registry table. For more information, see Specifying a Primary Key for the Registry Table.

Multi-Line String Value Dialog Box

The Multi-Line String Value dialog box enables you to modify a selected registry value by typing a line for each null-delimited string.

Task ***To enter a multi-line string value in the Multi-Line String Value dialog box:***

1. Choose how you want to modify the registry value. You can choose from the following options:
   - Append
   - Prepend
   - Replace

2. In the grid part of the dialog box, type a line for each null-delimited string or modify it. Right-click the grid to display the context menu or press the following keys in the grid below associated with the following actions:

Table 9-19 ▪ Context-Menu Commands for Entering Strings in the Grid

Action	Shortcut Key
Add String	Ins
Rename String	F2
Delete String	Del
Move Up	
Move Down	

3. Click **OK** when you have entered a line for each null-delimited string. The strings are automatically concatenated.

Note ▪ *Strings can contain only spaces, but they cannot be empty or [~], which is the delimiter for the strings.*

New Project Dialog Box

The New Project dialog box opens when you are creating a new project in InstallShield. In this dialog box, you can select a project type, name your project, and provide a location for the project's files. After you select a project type and click OK, your project opens in the InstallShield Installation Development Environment (IDE).

Note ▪ *If you select Express Project, the Project Assistant is launched to help you create your project.*

Table 9-20 ▪ New Project Dialog Box Settings

Setting	Description
Project Name	Type a name for your project in this field.
Project Language	Select the language that should be used for your installation project.
Location	Type a location or click Browse to navigate to a project location. To change the default project location displayed, change the Project Location path, which is located on the File Locations tab in the Options dialog box.
Create project file in 'Project Name' subfolder	Select this option if you want InstallShield to create a subfolder with your project's name in your Projects location.

***Edition** ▪ InstallShield Premier and InstallShield Professional provide additional functionality in creating new projects. For example, the Premier edition allows you to have multiple languages in the same installation project.*

Options Dialog Box

The Options dialog box enables you to specify preferences for creating projects and working in the InstallShield Installation Development Environment (IDE). To view and edit these global settings, on the Tools menu, click Options.

The Options dialog box is organized into multiple task-related tabs:

- General
- File Locations
- Preferences
- Merge Module Options
- Quality
- Updates
- .NET
- Files View
- File Extensions
- Prerequisites

General Tab

The General tab on the Options dialog box is where you specify preferences for the help system and for build errors.

Table 9-21 ▪ General Tab Settings

Setting	Description
Help window on top	If you would like the help window to remain on top of the Installation Development Environment (IDE), select this check box. If you want the help window to fall to the background when you click in the IDE, clear this check box.
Stop build process when first error is encountered	To abort the build process when a build error occurs, select this check box.

File Locations Tab

The File Locations tab on the Options dialog box is where you set the default directory for your project files.

Table 9-22 ▪ File Locations Tab Settings

Setting	Description
Project Location	In this field, type the path or browse to the location for your installation project files. This location is the default folder for new installation projects. All of your source and release files, such as your project (.ise) file, installation package (.msi file), and disk image files, are stored in subfolders of your project location. By default, your project is stored in the *InstallShield 2022 Projects* folder at the root level of your hard drive.

Preferences Tab

The Preferences tab on the Options dialog box is where you specify preferences for run commands and for project reloads.

Table 9-23 ▪ Preferences Tab Settings

Setting	Description
Uninstall before installing	If you would like InstallShield to automatically uninstall your product before you rerun it by relaunching it from the Build menu, select this check box.
Reload last opened project on startup	If you would like InstallShield to automatically reload the most recently opened project when you launch InstallShield, select this check box.

Merge Module Options Tab

The Merge Module Options tab on the Options dialog box is where you specify preferences for merge module locations and file searching.

Table 9-24 ▪ Merge Module Options Tab Settings

Setting	Description

Merge Module Locations (Current User) and Merge Module Locations (All Users)	Enter the paths where you store merge modules (.msm files). Separate additional paths with a comma, as in the following example:
	`C:\MergeModules,C:\My Files\MergeModules`
	Note that you can use path variables in the path, as in the following example:
	`<ISProductFolder>\Modules\i386,<ISProjectFolder>\MyCustomModules`
	You can also use environment variables in the path.
	The All Users option is available if you want to run a command-line build under a system account for which you cannot easily update the user settings.
	InstallShield provides additional ways for specifying the folders that contain merge modules. For more information, see Specifying the Directories that Contain Merge Modules.
	
	Note ▪ *In order to configure the All Users option, you must be running InstallShield with administrative privileges. If you do not have administrative privileges, this option is disabled. To learn more, see Launching InstallShield with vs. Without Administrative Privileges.*

Matching files must have the same version	When you add a file to your project, InstallShield searches the merge modules to see if there is a module that contains that file and notifies you of any matches. If the files must be the same version in order to be considered a match, select this check box.
Matching files must have the same destination	When you add a file to your project, InstallShield searches the merge modules to see if there is a module that contains that file and notifies you of any matches. If the files must have the same destination in order to be considered a match, select this check box.

.NET Tab

The .NET tab on the Options dialog box is where you specify preferences for .NET projects. It is also where you specify the location of the `Regasm.exe` and `InstallUtilLib.dll` files, which are utilities that are included with the .NET Framework. These utilities are used for COM interop and .NET custom actions.

Table 9-25 ▪ .NET Tab Settings

Setting	Description
Default .NET Scan At Build File Setting	In this list, select how the .NET Scan at Build setting should be configured for new portable executable files that are added to your project.
.NET Framework File Locations	`Regasm.exe` and `InstallUtilLib.dll` are utilities that are included with each version of the .NET Framework. Specify the path for the directory that contains the version of these files that you want to use at build time for releases that include .NET installer classes and COM interop. • **32-Bit Location**—Type the path or browse to the location of `Regasm.exe` and `InstallUtilLib.dll`. • **64-Bit Location**—This option is disabled if you are using InstallShield on a 32-bit system. If you are using InstallShield on a 64-bit system, type the path or browse to the location of `Regasm.exe` and `InstallUtilLib.dll`. ***Note ▪*** *In order to configure these options, you must be running InstallShield with administrative privileges. If you do not have administrative privileges, this option is disabled. To learn more, see Launching InstallShield with vs. Without Administrative Privileges.*

Files View Tab

In the Files View tab on the Options dialog box, you can indicate preferences such as which columns you want to be displayed in various areas of InstallShield.

If you make any changes on this tab, close and reopen your project to see the changes take effect.

Table 9-26 ▪ Files View Tab Settings

Setting	Description
Size	To include a Size column in the following areas of InstallShield, select this check box: • Files and Features view • Files view • Application Files page of the Project Assistant

Table 9-26 ▪ Files View Tab Settings (cont.)

Setting	Description
Link To	To include a Link To column in the following areas of InstallShield, select this check box: • Files and Features view • Files view • Application Files page of the Project Assistant
Modified Date	To include a Modified Date column in the following areas of InstallShield, select this check box: • Files and Features view • Files view • Application Files page of the Project Assistant
Destination	To include a Destination column in the Application Files page of the Project Assistant, select this check box.
Version	To include a Version column in the following areas of InstallShield, select this check box: • Files and Features view • Files view ***Important ▪*** *Selecting the Version check box will slow the performance of the aforementioned views.*
File Key	To include a Key column in the Files view, select this check box.
Maximum files to display	Specify the maximum number of files that you want to be displayed for a dynamic file link in the following areas of InstallShield: • Files and Features view • Files view • Application Files page of the Project Assistant The default value is 1000. If the dynamic file link includes more than the specified number of files, the top entry in the list of files is called ***List Truncated***.

File Extensions Tab

The File Extensions tab on the Options dialog box is where you specify preferences for Portable Executable (PE) files. InstallShield refers to this list of PE files when it creates components for your project. (A component is the smallest installable part of a product. The Express edition of InstallShield creates components for you automatically. For more information, see Installation Fundamentals.) For example:

- When you add a PE file to a folder in the **Destination computer's folders** pane in the Files view, InstallShield creates a new component for it and sets it as the component's key file.
- When you use the best practice method for creating dynamic file links, InstallShield creates at build time a separate component for each PE file in the dynamically linked folder. Each PE file is the key file of its component. To learn more about best practice dynamic file linking, see Determining the Appropriate Component Creation Method for Dynamically Linked Files.

The following setting is available on this tab:

Table 9-27 ▪ File Extensions Tab Settings

Setting	Description
Portable Executable File Extensions	In this box, type the file extensions that you would like InstallShield to consider to be PE files. Separate extensions with a comma. The default entry for this box is the following: AX,EXE,DLL,OCX,VXD,CHM,HLP,TLB

Prerequisites Tab

The Prerequisites tab on the Options dialog box enables you to set preferences for InstallShield prerequisites.

Table 9-28 ▪ Prerequisites Tab Settings

Setting	Description

Prerequisite File Locations (Current User) and Prerequisite File Locations (All Users)	Specify the paths for the folders where you store InstallShield prerequisite files (.prq files).
	To specify more than one location, separate each path with a comma, as in the following example:
	`C:\Prerequisites,C:\My Files\Prerequisites`
	Note that you can use path variables in the path, as in the following example:
	`<ISProductFolder>\SetupPrerequisites,<ISProjectFolder>\MyCustomPrerequisites`
	The All Users option is available if you want to run a command-line build under a system account for which you cannot easily update the user settings.
	InstallShield provides additional ways for specifying the folders that contain InstallShield prerequisite files. For more information, see Specifying the Directories that Contain InstallShield Prerequisites.
	
	Note ▪ *In order to configure the All Users option, you must be running InstallShield with administrative privileges. If you do not have administrative privileges, this option is disabled. To learn more, see Launching InstallShield with vs. Without Administrative Privileges.*

Outputs Dialog Box

The Outputs dialog box displays information about a project output group in the File System Editor. This dialog box is available for InstallShield projects created in Microsoft Visual Studio.

Task ***To access the Outputs dialog box, do one of the following:***

- Select the **Outputs** property in the **Properties** window when a project output group is selected in the **File System Editor**.
- In the **Files** view, right-click an item in the **Source computer's files** pane and click **Resolve Project Output**.
- In the Files view, right-click an item in the **Destination computer's files** pane and click **Resolve Project Output**.

Note ▪ *If multiple project output groups are selected, information is displayed only for the first group selected.*

Dialog Box Settings

Target Name

Displays the file name for the selected project output group as it will be displayed on a target computer. This field is read only.

Source Path

Displays the path to the project output group files on the development computer. This field is read only.

Permissions Dialog Boxes for Files and Directories

Permissions Dialog Box

The Permissions dialog box lets you configure settings for securing files and folders for end users who run your product in a locked-down environment. You can assign permissions for a file or folder to specific groups and users. For example, you may assign Read, Write, and Delete permissions for a particular file to the Administrators group, but only Read permissions for all of the users in a different group.

Depending on what is selected for the Locked-Down Permissions setting in the General Information view of your project, InstallShield adds permissions data to either the `ISLockPermissions` table or the `LockPermissions` table. To learn more, see Securing Files, Folders, and Registry Keys in a Locked-Down Environment.

The following table describes the different areas on the Permissions dialog box.

Table 9-29 ▪ Areas of the Permissions Dialog Box

Area	Description
Name(s)	In this grid, you can enter any combination of domain and user names. To add an entry, right-click the grid and click New. You can modify or delete entries using the same context menu. To specify the current user's domain, select **[%USERDOMAIN]** in the Domain field. To specify the user that is currently running the installation, select **[LogonUser]** in the User field. To set permissions for user accounts on a local system, leave the Domain field blank. Note that you can enter in the Domain or User fields any Windows Installer property that is set at run time—for example, **[MYPROPERTY]**. If the custom InstallShield handling option is selected for the Locked-Down Permissions setting in the General Information view, the User field contains a list of well-known security identifiers (SIDs). Most of the SIDs are not supported by the traditional Windows Installer handling option. The custom InstallShield handling option supports localized names for all of the SIDs that are listed in the User field. With the traditional option, if you try to use a localized name to set permissions on a non-English system, the installation may fail. ***Tip ▪*** *For more information on the custom InstallShield handling option and the traditional Windows Installer handling option, see Securing Files, Folders, and Registry Keys in a Locked-Down Environment.*
Permissions	Select a name in the Name(s) area, and then select or clear the check boxes in the Permissions box to configure the corresponding permissions for the file or folder. Once you have selected a permission, you can click the Advanced button to specify other associated permissions and advanced settings.
Deny Access	If you want to explicitly deny the permissions that you are selecting in the Permissions box, select this check box. This check box is available only if the custom InstallShield handling option is selected for the Locked-Down Permissions setting in the General Information view. The traditional Windows Installer handling option does not include support for this behavior. ***Tip ▪*** *For more information on the custom InstallShield handling option and the traditional Windows Installer handling option, see Securing Files, Folders, and Registry Keys in a Locked-Down Environment.*

Advanced Permissions Dialog Box

If you click the Advanced button on the Permissions dialog box, the Advanced Permissions dialog box opens. The following table describes the different areas on the Advanced Permissions dialog box.

Table 9-30 ▪ Areas of the Advanced Permissions Dialog Box

Area	Description
Advanced Permissions	In this box, select the check boxes for the permissions that you want to set.
Apply these permissions to child objects	If you are configuring permissions for a folder and you want the permissions to be applied to all of the folder's subfolders and files, select this check box. This check box is available only if the custom InstallShield handling option is selected for the Locked-Down Permissions setting in the General Information view. The traditional Windows Installer handling option does not include support for this behavior. ***Tip ▪*** *For more information on the custom InstallShield handling option and the traditional Windows Installer handling option, see Securing Files, Folders, and Registry Keys in a Locked-Down Environment.*

Permissions Dialog Boxes for Registry Keys

Permissions Dialog Box

The Permissions dialog box lets you configure settings for securing registry keys for end users who run your product in a locked-down environment. You can assign permissions for a registry key to specific groups and users. For example, you may assign Read, Write, and Delete permissions for a particular registry key to the Administrators group, but only Read permissions for all of the users in a different group.

Depending on what is selected for the Locked-Down Permissions setting in the General Information view of your project, InstallShield adds permissions data to either the `ISLockPermissions` table or the `LockPermissions` table. To learn more, see Securing Files, Folders, and Registry Keys in a Locked-Down Environment.

The following table describes the different areas on the Permissions dialog box.

Table 9-31 ▪ Areas of the Permissions Dialog Box

Area	Description
Name(s)	In this grid, you can enter any combination of domain and user names. To add an entry, right-click the grid and click New. You can modify or delete entries using the same context menu. To specify the current user's domain, select **[%USERDOMAIN]** in the Domain field. To specify the user that is currently running the installation, select **[LogonUser]** in the User field. To set permissions for user accounts on a local system, leave the Domain field blank. Note that you can enter in the Domain or User fields any Windows Installer property that is set at run time—for example, **[MYPROPERTY]**. If the custom InstallShield handling option is selected for the Locked-Down Permissions setting in the General Information view, the User field contains a list of well-known security identifiers (SIDs). Most of the SIDs are not supported by the traditional Windows Installer handling option. The custom InstallShield handling option supports localized names for all of the SIDs that are listed in the User field. With the traditional option, if you try to use a localized name to set permissions on a non-English system, the installation may fail. ***Tip ▪*** *For more information on the custom InstallShield handling option and the traditional Windows Installer handling option, see Securing Files, Folders, and Registry Keys in a Locked-Down Environment.*
Permissions	Select a name in the Name(s) area, and then select or clear the check boxes in the Permissions box to configure the corresponding permissions for the registry key. Once you have selected a permission, you can click the Advanced button to specify other associated permissions and advanced settings.
Deny Access	If you want to explicitly deny the permissions that you are selecting in the Permissions box, select this check box. This check box is available only if the custom InstallShield handling option is selected for the Locked-Down Permissions setting in the General Information view. The traditional Windows Installer handling option does not include support for this behavior. ***Tip ▪*** *For more information on the custom InstallShield handling option and the traditional Windows Installer handling option, see Securing Files, Folders, and Registry Keys in a Locked-Down Environment.*

Advanced Permissions Dialog Box

If you click the Advanced button on the Permissions dialog box, the Advanced Permissions dialog box opens. The following table describes the different areas on the Advanced Permissions dialog box.

Table 9-32 ▪ Areas of the Advanced Permissions Dialog Box

Area	Description
Advanced Permissions	In this box, select the check boxes for the permissions that you want to set.
Apply these permissions to child objects	If you are configuring permissions for a registry key and you want the permissions to be applied to all of the key's subkey, select this check box. This check box is available only if the custom InstallShield handling option is selected for the Locked-Down Permissions setting in the General Information view. The traditional Windows Installer handling option does not include support for this behavior. ***Tip ▪*** *For more information on the custom InstallShield handling option and the traditional Windows Installer handling option, see Securing Files, Folders, and Registry Keys in a Locked-Down Environment.*

Select Icon Dialog Box

The Select Icon dialog box enables you to browse for an icon in .ico files, .dll files, and .exe files. This simplifies the process of specifying the location of the icon file and the index of the icon within the file.

Task ***To select an icon:***

1. Locate a file to view by entering the path to the file in the **File name** box and pressing ENTER, or by clicking **Open Folder** and browsing to the file. All the icons within the file are displayed in the list.
2. When you locate an icon file, you can select the icon to be used with the mouse or the cursor keys. Use the **View large icons** and **View small icons** options to preview the icons in the list and see how they will appear in the standard 32x32 and 16x16 sizes.

Note ▪ *The View large icons and View small icons radio buttons have no effect on the index of the icon selected. They are used only to preview the icons in each size.*

Set INSTALLDIR Dialog Box/Set DATABASEDIR Dialog Box

Use these dialog box to browse to a directory, create a new directory, rename a directory, or delete a directory. When you click OK, either `INSTALLDIR` or `DATABASEDIR` is set to the selected directory—depending on the dialog box you are in.

Dialog Box Settings

Destination Directories

This field lists all of the currently available destination directories. You can select, create, rename, or delete directories in this field.

Selecting a Directory

Task ***To select a directory:***

1. Click a directory to select it.
2. Click **OK**.

Creating a New Directory

Task ***To create a new directory***

1. Type the directory name.
2. Provide a directory identifier, if necessary.

Renaming a Directory

Task ***To rename a directory:***

1. Select a directory or **Destination Computer** and press F2, or right-click and click **Rename**.
2. Type the new directory name. Note that you cannot rename predefined directories.
3. Rename the directory identifier, if necessary, to be consistent with the directory's new name.

Deleting a Directory

Task ***To delete a directory:***

Select a directory and press DELETE, or right-click the directory and click **Delete**. Note that you cannot delete predefined directories.

Note ▪ *When you delete a directory, any subdirectories beneath the selected directory are also deleted.*

Directory Identifier

You can use the Directory Identifier field to provide a user-friendly name for a directory. For example, if you have a directory—`ProgramFilesFolder\MyProgram\Graphics\Jpg`—you can use just **JPG** to identify the directory path. The InstallShield interface displays the path as follows:

```
{JPG} [ProgramFilesFolder]MyProgram\Graphics\Jpg
```

Note ▪ *The directory identifier must be a valid MSI identifier.*

Settings Dialog Box

The Settings dialog box enables you to set preferences for how InstallShield builds the installation for the project that is currently open.

Task ***To open the Settings dialog box:***

On the **Build** menu, click **Settings**.

The following tab is available on the Settings dialog box:

- MSI Log File tab (Windows Installer projects only)

MSI Log File Tab

The MSI Log File tab on the Settings dialog box enables you to create a log file that you can use when you are running your installation from within InstallShield. In addition, this tab lets you select the type of information that you would like written to the log file.

Table 9-33 ▪ MSI Log File Tab Settings

Setting	Description
MSI Log File Options	Select the check boxes for the types of messages that you want to be displayed in your log file. When you select a check box, the corresponding argument is added to the MsiExec.EXE Command-Line Arguments box.
MsiExec.EXE Command-Line Arguments	Enter the command-line arguments that you would like to pass to `MsiExec.exe` when your installation is run. When check boxes are selected in the MSI Log File Options area, the corresponding command-line parameters are added to this box.
Log File	Type the name of the log file that will be created when your installation is run.

System Hardware Requirements Dialog Box

The System Hardware Requirements dialog box enables you to set operating system requirements for your product. If a target system does not meet the operating system requirements that you have set, the installation does not allow your product to be installed.

When you are setting operating system requirements through this dialog box, do one of the following:

- If you do not want to set an operating system condition for your product, select the **All Operating Systems** check box.
- If you do want to limit the list of supported operating systems, clear the **All Operating Systems** check box. Then select the check boxes that correspond with the operating systems that your product supports.

***Tip** ▪ When you specify operating system requirements for your product, you are essentially excluding operating systems that do not support your product.*

For example, if you select only the check box for the latest Windows operating system, InstallShield creates a launch condition to exclude the operating systems that you did not select in the Requirements view. With this type of launch condition, future versions of Windows operating systems are supported automatically because they are not excluded in the launch condition.

Update Merge Module Search Path Dialog Box

This dialog box appears when you use the **Browse for Merge Module** command to add a merge module to your installation project. It displays the path that will be added to the merge module search path (Merge Module Locations field) in the Options dialog.

Click OK to approve the action.

Upgrade Express Project Name Dialog Box

After you have selected the InstallShield Express 2.x project that you want to import into the current version of InstallShield, you are be prompted to enter the path and name of the new project file you want to create.

Enter the fully qualified file name, or click the Browse button to navigate to the folder where the file will be created.

Wizard Reference

InstallShield includes many wizards to assist you in creating your installation project.

Project ▪ *Not all wizards are available with all project types.*

The following wizards are available in InstallShield.

- Create New QuickPatch Wizard
- DirectX Object Wizard
- Dynamic Scanning Wizard
- Export String Table Wizard
- Import REG File Wizard
- Import String Table Wizard
- Redistributable Downloader Wizard
- Static Scanning Wizard
- System Search Wizard
- Visual Studio .NET Wizard for Visual Basic .NET, Visual C++ .NET, and C# .NET
- Visual Studio Deployment Project Import Wizard
- Web Deployment Wizard

Create New QuickPatch Wizard

A QuickPatch project is recommended for installation authors who want to ship small, single updates to their users. The creation of a QuickPatch project begins with the Create New QuickPatch Wizard.

The following panels are associated with the Create New QuickPatch Wizard:

- Welcome
- Project Name
- Release to Patch
- Final

Welcome Panel

A QuickPatch project is recommended for setup authors who want to ship small, single updates to their users. Running the Create New QuickPatch Wizard produces the following deliverable types: *.msp and *.exe files.

The creation of a QuickPatch project always begins with the Create New QuickPatch Wizard. Following through the wizard will ensure that all basic requirements for the QuickPatch project are met.

Click Next to continue.

Note ▪ *Anytime you click Cancel or Exit in the wizard, QuickPatch project creation will be incomplete, and the QuickPatch project will not open in InstallShield.*

Project Name Panel

Since QuickPatch projects rely on the setup author's knowledge of what to change or patch in the original installation or `Setup.exe` file, you must specify the path to the original installation. The original setup is the base setup for the new patch. The Create New QuickPatch Wizard will create a patch that can only update this installation.

In this panel, configure the following settings:

Table 9-1 ▪ Project Name Panel Settings

Setting	Description
QuickPatch Project Name	Enter a name for your QuickPatch Project or click the browse button to open an existing project. If applicable, you will be prompted to verify whether or not you want to overwrite an existing project.
Built MSI package or Setup.exe to patch	Specify the location of your built MSI package or Setup.exe or click the browse button to locate an existing setup. The Create New QuickPatch Wizard will create a project that can only validate this setup.

Release to Patch Panel

In this panel, select a release in a project from which to build the patch. You can only select one release which has been built. You also have the option to browse for an external release. To activate that option, click the associated radio button, and browse for the release.

Final Panel

You will see this wizard panel after you have successfully completed the Create New QuickPatch Wizard. When you see this panel, an administrative installation of your project is about to be launched shortly. Once the administrative installation is complete, InstallShield creates a QuickPatch project with the information you provided in the wizard panels.

DirectX Object Wizard

The DirectX Object Wizard enables you to include the Microsoft DirectX 9c redistributable files in your installation project and to set several options.

Note ▪ *For information about which files are included with the DirectX object, see Including the DirectX 9.0 Object.*

The optional ManagedDX component of DirectX 9c requires that the .NET Framework version 1.1 or later be installed on the system.

The following wizard panels are associated with the DirectX Object Wizard:

- Welcome
- Object Settings
- Summary

For more information about DirectX, see the latest DirectX SDK or Microsoft's MSDN Web site.

Welcome Panel

The DirectX Object Wizard enables you to include the Microsoft DirectX 9c redistributable files in your installation project and to set several options.

Note ▪ *For information about which files are included with the DirectX object, see Including the DirectX 9.0 Object.*

The optional ManagedDX component of DirectX 9c requires that the .NET Framework version 1.1 or later be installed on the system.

For more information about DirectX, see the latest DirectX SDK or Microsoft's MSDN Web site.

Object Settings Panel

The Object Settings panel enables you to set the following options:

Table 9-2 ▪ Object Settings Panel Options

Option	Description
Place DirectX files in a folder in Disk1	If you select this check box, InstallShield creates a DirectX folder for your installation at build time and places it in the Disk1 folder of your release. If you clear this check box, InstallShield streams the DirectX redistributable files into the .msi file. If you are creating a single-file executable installation, clear this check box so that the files are included in the .msi file. For information on the DirectX files that are included in the object, see Including the DirectX 9.0 Object.
Show the Microsoft DirectX EULA before starting the DirectX 9 installation	By default, the Microsoft DirectX EULA is displayed to the end user before the DirectX 9 installation begins. To prevent the EULA from being displayed, clear this check box.

For more information about DirectX, see the latest DirectX SDK or Microsoft's MSDN Web site.

Summary Panel

The Summary panel enables you to review the options that you configured on the Object Settings panel. Click Finish to accept the settings and add the DirectX 9 redistributable files to your project.

Dynamic Scanning Wizard

The Dynamic Scanning Wizard is an easy-to-use tool that monitors your system while an executable file runs. The wizard displays a list of .dll and .ocx files that may be required by the executable file, and it lets you specify whether you want to include each one in your project. The wizard can scan for an executable file that is already included in your project, or you can use the Specify the Executable panel in the wizard to select a new executable file that you want to scan and add to your project prior to the scanning process.

Task ***To launch the Dynamic Scanning Wizard:***

1. In the View List under **Specify Application Data**, click **Dependencies**.
2. Click the **Perform Dynamic Scanning** button.

The following panels are associated with the Dynamic Scanning Wizard:

- Welcome
- Filter Files
- Specify the Executable
- Specify Application File
- Launch the Application
- Your Application is Running
- File Selection
- Scan Results
- Completing the Dynamic Scanning Wizard

Welcome Panel

The Dynamic Scanning Wizard provides you with an easy path to add your application's dependency files to your project. Before you begin using the scanner, it is recommended that you add the target executable file to your project.

Click Next to begin using this wizard.

Filter Files Panel

The Dynamic Scanning Wizard may list as dependencies certain files that you do not want to add to your installation. For example, common system files that are already present on target machines usually do not need to be reinstalled. To avoid having these files added to your project when you run the scanner, select the **Filter files** check box on the Filter Files panel.

To learn how to customize the list of files that are excluded from scans, see Filtering Files in Dependency Scanners.

Specify the Executable Panel

The Specify the Executable panel is where you specify whether you want to scan an executable file that is already included in your project, or one that has not yet been added.

Table 9-3 ▪ Specify the Executable Panel Settings

Setting	Description
I would like to select an executable file from my project	If the executable file that you would like to scan has already been added to your project, select this option.
I would like to select a new executable	If the executable you want to scan is not currently included in your project, select this option.

Specify Application File Panel

The Specify Application File panel is where you to select the specific executable file (.exe) that you would like to scan. Additionally, you can specify command-line parameters for the file and a working folder.

Table 9-4 ▪ Specify Application File Panel Settings

Setting	Description
Application	Specify which executable file you would like to scan. If you choose to scan a file that is already included in your project, select it from the list of executable files that present in your project. If you choose to scan a file that is not yet a part of your project, enter the path to that file or click the Browse button to navigate to it.
Command Line	Enter the command-line parameters that you would like to pass to the executable file. These parameters are used only during the scanning process; they are not used after the wizard has been dismissed.
Working Folder	Enter the path to the working folder for this application, or click the Browse button to navigate to the directory. By default, this directory is set to the same folder in which the application you choose to scan resides.

Launch the Application

Before the Dynamic Scanning Wizard begins scanning your application, it must launch the application. After your application is launched by the wizard, you should use as many of the application's menu items and features as you can. This helps to identify where dependency files are located so that they can be added to your project.

Click Next to launch your application and begin scanning for dependencies.

Your Application Is Running Panel

The Your Application is Running panel is displayed while your application is running. When you have finished using your application, click the Done button to view the results of the scan. No files are added until you have confirmed the findings of the scan.

File Selection Panel

The File Selection panel displays a list of possible files and merge modules that you may need to add to your project. Use this panel to select the ones that you want to include in your installation. For more information, see Reviewing Dependency Scanner Results.

Table 9-5 ▪ File Selection Panel Settings

Setting	Description
File	Indicate which files and merge modules you want to add to your project by selecting the appropriate check boxes.
Deselect All	If you want to clear all of the check boxes, click this button. You can then manually select each check box that corresponds with a file or merge module that you would like to add to your project.
Select All	If you want to select all of the check boxes, click this button. You can then manually clear each check box that corresponds with a file or merge module that you do not want to add to your project.

Scan Results Panel

The Scan Results panel shows the dependencies that the wizard identified and that you selected to be added to your project.

To add these dependencies to your project, click the Next button. To exit the wizard without adding the dependencies, click the Cancel button. To review the list of potential dependencies again and add or remove any of them, click the Back button.

Completing the Dynamic Scanning Wizard Panel

When the Dynamic Scanning Wizard shows the Completing the Dynamic Scanning Wizard panel, the wizard has added the executable file's dependencies to your project. If you chose to scan an executable file that was not already included in your project, that .exe file has been added as well.

Click Finish to close the wizard and return to InstallShield.

Export String Table Wizard

The Export String Table wizard allows you to export your string table to a text (.txt) file, which can then be imported back into InstallShield when translation has been completed. The following panels are associated with the Export String Table wizard:

- Welcome
- File Name
- Wizard Complete

Welcome Panel

The easiest way to translate the run-time strings of your setup is to export them to a text (.txt) file and then have them translated. Once the translation is complete you can import those strings back into your setup project for a localized version of your installation. The Export String Table wizard walks you through the task of exporting all of your strings to a text file.

File Name Panel

This panel allows you to specify the location and filename of the text file all of your strings will be exported to.

Table 9-6 ▪ File Name Panel Settings

Setting	Description
File Name	Enter the path and filename of the text (.txt) file you want all of your strings sent to, or click the Browse button to navigate to this file.

Wizard Complete Panel

At this time all of your strings have been sent to the file you specified in the previous panel. Click Cancel to discard your changes or click Finish to have the changes saved and return to the IDE.

Import REG File Wizard

The Import REG File Wizard allows you to import existing registry data (.reg) into your InstallShield project. This registry data is added to the target system's registry during the setup project.

The following panels are associated with the Import REG File wizard:

- Welcome
- Import Registry File
- Import Conflict Options
- Import Progress

Welcome Panel

The Import REG File Wizard allows you to import existing registry data (.reg) into your InstallShield project. This registry data is added to the target system's registry during the setup project.

Note ▪ *Before you begin importing registry data be sure that you selected the proper feature to which you want this data added. To specify a feature, cancel the wizard and select the appropriate feature from the Feature list at the top of the Registry view.*

Click Next to begin importing your .reg file.

Caution ▪ *InstallShield can only import .reg files created by exporting in Regedit, or files that follow that exact format. Additionally, InstallShield does not support multiline registry values.*

Import Registry File Panel

In this panel you can specify the .reg file you want to import.

Table 9-7 ▪ Import Regather File Panel Settings

Property	Description
Registry File	Enter the path to the .reg file you want to import, or click the Browse button to navigate to this file. ***Caution*** ▪ *InstallShield can only import .reg files created by exporting in Regedit, or files that follow that exact format. Additionally, InstallShield does not support multiline registry values.*

Import Conflict Options Panel

Because your setup may already contain registry data that could conflict with information stored within the .reg file you are importing, you can select how you want to handle any conflicts.

Table 9-8 ▪ Import Conflict Options Panel Settings

Setting	Description
Overwrite the registry data	Select this option if you want data stored within the REG file you are importing to overwrite any conflicting data already present in your installation project.
Do not overwrite the registry data	Select this option if you want to retain the data already present in your installation project when a conflict arises during the import progress. All nonconflicting registry data are still imported.

Import Progress Panel

This panel displays the import progress of the REG file. Click Cancel to stop the import or wait until the wizard finishes importing your REG file and click Finish to return to the IDE.

Import String Table Wizard

Use the Import String Table wizard to import a string table into InstallShield. This wizard walks you through the task of importing your strings back into InstallShield after you have had them translated. The following panels are associated with the Import String Table wizard:

- Welcome
- File Name
- Wizard Complete

Welcome Panel

If you are creating localized versions of your setup you will find it much easier to export all of your run-time strings for translation. Once those strings have been translated you can import them back into your setup project. The Import String Table wizard allows you to point to the text (.txt) file you want to import that contains all of your translated run-time strings.

When you import a string table, all of your existing strings are overwritten by those in the string table you are importing. Therefore, make sure you have saved a copy of your original strings before you import the translated version.

File Name Panel

This dialog allows you to point to the specific text (.txt) file you want to import.

Table 9-9 ▪ File Name Panel Settings

Setting	Description
File Name	Enter the path and filename of the text (.txt) file containing the strings you want to import, or click the Browse button to navigate to this file.

Click the Next button to import these strings into your setup project, overwriting any existing strings you may have.

Wizard Complete Panel

At this point the strings contained within the text file you specified have been imported into your setup project. If you want to exit the wizard without saving your changes, click the Cancel button. Click Finish to save your changes and return to the IDE.

Redistributable Downloader Wizard

The Redistributable Downloader Wizard allows you to quickly download third-party redistributables, merge modules, prerequisites, and other files to your local machine. You can launch the Redistributable Downloader Wizard from the Release Wizard's .NET Run-Time Options panel and from the Releases view (.NET 1.1/2.0 Core Language and .NET 1.1/2.0 Language Packs settings).

Task ***To use the Redistributable Downloader Wizard:***

1. In the **Select Files to Download** panel, select the redistributables\prerequisites that you want to download and click **Next** to move to the **Progress** panel.
2. When the progress bars complete, click **Next**.

The final panel displays the list of redistributables\prerequisites that the wizard downloaded to your machine. Click **Finish** to exit the wizard.

Static Scanning Wizard

The Static Scanning Wizard enables you to scan the files that are in your project for potential dependencies that they may require. This wizard scans all .exe, .dll, .ocx, .sys, .com, .drv, .scr, and .cpl files in your project and lets you add any detected dependencies to your installation.

The new files that added to your project are added to the same feature as the file that depends on them, thereby ensuring they get installed when needed.

Task ***To launch the Static Scanning Wizard:***

1. In the View List under **Specify Application Data**, click **Dependencies**.
2. Click the **Perform Static Scanning** button.

The following panels are associated with the Static Scanning Wizard:

- Welcome
- Filter Files
- Scanning Progress
- File Selection
- Scan Results
- Completing the Static Scanning Wizard

Welcome Panel

The Static Scanning Wizard enables you to scan the files that are in your project for potential dependencies that they may require. This wizard scans all .exe, .dll, .ocx, .sys, .com, .drv, .scr, and .cpl files in your project and lets you add any detected dependencies to your installation.

Click the Next button to begin scanning your project's files for dependencies.

Filter Files Panel

The Static Scanning Wizard may list as dependencies certain files that you do not want added to your installation. For example, common system files that are already present on target machines usually do not need to be reinstalled. To avoid having these files added to your installation when you run the scanner, select the **Filter files** check box on the Filter Files panel.

To learn how to customize the list of files that are excluded from scans, see Filtering Files in Dependency Scanners.

Scanning Progress Panel

The Scanning Progress panel is displayed when the Static Scanning Wizard is scanning all .exe, .dll, .ocx, .sys, .com, .drv, scr, and .cpl files in your project for dependencies. No files are added until you have confirmed the findings of the scan.

File Selection Panel

The File Selection panel displays a list of possible files and merge modules that you may need to add to your project. Use this panel to select the ones that you want to include in your installation. For more information, see Reviewing Dependency Scanner Results.

Table 9-10 ▪ File Selection Panel Settings

Setting	Description
File	Indicate which files you want added to your installation by selecting the appropriate check boxes. The meaning of each type of icon is as follows: • : This icon indicates that the file is a system or driver file. This type of file is normally not redistributed as part of an installation, and it can potentially cause destination machines to become inoperable. Ensure these files are necessary before including them. • : This icon indicates a non-system file. Select the check box to the left of the icon to include the file in your installation. • : This icon indicates a merge module. Select the check box to the left of the icon to add the specified merge module to your project. • : This is the icon for a .NET assembly in your installation project. • : This icon identifies a file that the Static Scanner has detected but that is already in the project.
Deselect All	If you want to clear all of the check boxes, click this button. You can then manually select each check box that corresponds with a file or merge module that you would like to add to your project.
Select All	If you want to select all of the check boxes, click this button. You can then manually clear each check box that corresponds with a file or merge module that you do not want to add to your project.

Scan Results Panel

The Scan Results panel shows the dependencies that the wizard identified and that you selected to be added to your project.

To add these dependencies to your project, click the Next button. To exit the wizard without adding the dependencies, click the Cancel button. To review the list of potential dependencies again and add or remove any of them, click the Back button.

Completing the Static Scanning Wizard Panel

When the Static Scanning Wizard shows the Completing the Static Scanning Wizard panel, the wizard has added the selected dependencies to your project.

Click Finish to close the wizard and return to InstallShield.

System Search Wizard

The Requirements view in InstallShield enables you to specify target system requirements in your installation project. When you specify system software requirements, you must run through the System Search Wizard. The System Search Wizard provides the Windows Installer capability to search for a particular file, folder, registry key or .ini value on a target system prior to installation.

Task ***To launch the System Search Wizard:***

1. In the View List under **Define Setup Requirements and Actions**, click **Requirements**.
2. In the explorer, right-click an item and click **Create New Launch Condition (System Search Wizard)**.

The System Search Wizard consists of the following panels:

- Welcome
- What do you want to find?
- How do you want to look for it?
- What do you want to do with the value?

Welcome Panel

The Welcome panel in the System Search Wizard. This wizard enables you to add or modify a system search in your project. Click Next to add or modify a system search.

What do you want to find? Panel

The **What do you want to find?** panel of the System Search Wizard is where you specify the type of item you are searching for and where to conduct that search on the target system. Available options are:

- File path, by searching folders
- Folder path, by searching all drives
- Folder path, by searching in a specific folder
- Folder path, by searching for a specific file
- File path, as specified by a registry entry
- Folder path, as specified by a registry entry
- Registry entry
- File path, as specified by an .ini file value
- Folder path, as specified by an .ini file value
- .ini file value

How do you want to look for it? Panel (Defining Your System Search Method)

The **How do you want to look for it?** panel contains settings that let you customize your search. The settings in this panel vary, depending on the type of search that you selected in the previous panel.

Table 9-11 ▪ Customizing Your Search

Type of Search	Corresponding Settings That Need to Be Configured
File path, by searching folders	Specify file details and where to search for the file. To learn more, see Specify the file details Panel (File Search Options).
Folder path, by searching all drives	Specify the name of the folder and how many subfolder levels you want to be searched. To learn more, see How do you want to look for it? Panel (Folder Search Options).
Folder path, by searching in a specific folder	Specify the name of the folder and where on target systems to look. To learn more, see How do you want to look for it? Panel (Specific Folder Options).
Folder path, by searching for a specific file	Specify file details and where to search for the file. To learn more, see Specify the file details Panel (File Search Options).
File path, as specified by a registry entry	Specify details about the registry entry. To learn more, see How do you want to look for it? Panel (Registry Search Options).
Folder path, as specified by a registry entry	Specify details about the registry entry. To learn more, see How do you want to look for it? Panel (Registry Search Options).
Registry entry	Specify details about the registry entry. To learn more, see How do you want to look for it? Panel (Registry Search Options).
File path, as specified by an .ini file value	Specify details about the .ini file value. To learn more, see How do you want to look for it? Panel (.ini File Search Options).
Folder path, as specified by an .ini file value	Specify details about the .ini file value. To learn more, see How do you want to look for it? Panel (.ini File Search Options).

Table 9-11 ▪ Customizing Your Search (cont.)

Type of Search	Corresponding Settings That Need to Be Configured
.ini file value	Specify details about the .ini file value. To learn more, see How do you want to look for it? Panel (.ini File Search Options).

Specify the file details Panel (File Search Options)

If you are configuring the **File path, by searching folders** type of system search, or the **Folder path, by searching for a specific file** type of search, the following settings are available on the **Specify the file details and specifically where to search for the file** panel:

Table 9-12 ▪ Specify the file details and specifically where to search for the file Panel Settings

Setting	Description
File Name	Enter the full name and extension of the file or application that you want to locate. ***Note ▪*** *When you enter a file name, the wizard enables the Details button. To modify your search to include any particular version, date, size, or language, click this button. To learn more, see File Details Dialog Box.*
Look In	Specify where on target systems you want Windows Installer to search.
Maximum number of subfolders to search	Specify the maximum number of subfolders that you want Windows Installer to search for the file on the target system.

How do you want to look for it? Panel (Folder Search Options)

If you are configuring the **Folder path, by searching all drives** type of system search, the following settings are available on the **How do you want to look for it?** panel:

Table 9-13 ▪ How do you want to look for it? Panel Settings

Setting	Description
Folder Name	Enter the full name and extension of the folder that you want Windows Installer to find.
Look In	Specify where on target systems you want Windows Installer to search.
Number of subfolder levels to search	Specify the number of subfolder levels that you want Windows Installer to search on target systems.

How do you want to look for it? Panel (Specific Folder Options)

If you are configuring the **Folder path, by searching in a specific folder** type of system search, the following settings are available on the **How do you want to look for it?** panel:

Table 9-14 ▪ How do you want to look for it? Panel Settings

Setting	Description
Folder Name	Enter the full name and extension of the folder you want to locate.
Look In	Specify where on target systems you want Windows Installer to search. You can either specify a full path or select a path from a previous search. If you want to specify a full path, you can click the Browse button to select an existing directory or create a new one. If you want to specify a path from a previous search, select it from the drop-down list. Note that the list is empty if your project does not contain any other searches.
Number of subfolder levels to search	Specify the number of subfolder levels that you want Windows Installer to search on target systems.

How do you want to look for it? Panel (Registry Search Options)

If you are configuring the **Folder path, as specified by a registry entry** type of system search, the **Folder path, as specified by a registry entry** type of system search, or the registry entry type of system search, the following settings are available on the **How do you want to look for it?** panel:

Table 9-15 ▪ How do you want to look for it? Panel Settings

Setting	Description
Registry Root	Select the registry root where you want Windows Installer to search on target systems.
Registry Key	Enter the exact registry key that is associated with the item that you are locating. For example, to locate `HKEY_LOCAL_MACHINE\SOFTWARE\Adobe\Acrobat` on the target system, you would enter **`SOFTWARE\Adobe\Acrobat`** in this setting. ***Tip ▪*** *Ensure that the syntax is correct by copying the correct key name from the Windows Registry Editor.*
Registry Value (Optional)	To search for a specific registry value, enter the registry value exactly as it appears in the Windows Registry Editor. ***Note ▪*** *If this setting is blank, the system search looks for the registry key's default value.*

Table 9-15 ▪ How do you want to look for it? Panel Settings (cont.)

Setting	Description
Search the 64-Bit Portion of the Registry	A 64-bit target system typically has two HKEY_LOCAL_MACHINE\Software keys: • HKLM\Software, which is for 64-bit applications • HKLM\Software\Wow6432Node, which is for 32-bit applications If you want Windows Installer to check the 64-bit portion of the registry (HKLM\Software) on 64-bit target systems, select this check box.

How do you want to look for it? Panel (.ini File Search Options)

If you are configuring the **File path, as specified by an .ini file value** type of system search, the **Folder path, as specified by an .ini file value** type of system search, or the **.ini file value** type of system search, the following settings are available on the **How do you want to look for it?** panel:

Table 9-16 ▪ How do you want to look for it? Panel Settings

Setting	Description
INI File Name	Specify the .ini file name as it should appear on target systems. The file must exist in the Windows folder.
INI Section Name	Specify the name of the section in the .ini file that contains the value that you want Windows Installer to find.
INI Key Name	Specify the name of the key that contains the value that you want Windows Installer to find..
Read entire line	If you want Windows Installer to read the entire line in the .ini file, select this check box.

What do you want to do with the value? Panel

In the **What do you want to do with the value?** panel, specify whether you want the installation to proceed if the search condition is found or not found.

This panel also is where you can specify text that you want to display when the installation is halted as a result of the search condition not being met.

Visual Studio .NET Wizard for Visual Basic .NET, Visual C++ .NET, and C# .NET

The Visual Studio .NET Wizard creates a new InstallShield installation project and adds it to a Microsoft Visual Studio .NET solution.

Note ▪ *The Visual Studio .NET Wizard is available only if Microsoft Visual Studio .NET is installed on your system.*

When you are creating the InstallShield installation project, the Visual Studio .NET Wizard does the following:

- Creates a new InstallShield project (with the file name specified on the New Project dialog box) and adds it to the solution (.sln file).
- Adds all dependencies to your project at build time if you have the Scan at Build option set to Dependencies and Properties (on the .NET tab of the Options dialog box).
- Adds the primary outputs from every project in the solution to the InstallShield project.
- Updates the release settings to deploy the appropriate version of the .NET Framework via download.

To launch the wizard, click the appropriate icon in the New Project dialog box.

Welcome Panel

The Visual Studio .NET Wizard allows you to add an InstallShield installation project to a Microsoft Visual Studio .NET solution.

The Visual Studio .NET Wizard is available only if Microsoft Visual Studio .NET is installed on your system.

Click Next to select a solution.

Solution Panel

In the Solution panel, type the path or browse to the Visual Studio .NET solution to which you want to add the InstallShield installation project.

Click Finish to create a new InstallShield project and add it to the selected solution.

Visual Studio Deployment Project Import Wizard

The Visual Studio Deployment Project Import Wizard enables you to import a Visual Studio setup or merge module project (.vdproj) into an InstallShield project (.ise). You can reuse this wizard if you want to import multiple Visual Studio projects into your InstallShield project.

Important ▪ *If the Visual Studio setup or merge module project that you want to import into an InstallShield project contains one or more project outputs, the InstallShield project must be in the same Visual Studio solution that contains the Visual Studio setup or merge module project and all of its project dependencies.*

Task ***To launch the Visual Studio Deployment Project Import Wizard, do one of the following:***

- If you are using InstallShield by itself (without integration with Visual Studio): On the **Project** menu, click **Visual Studio Deployment Project Import Wizard**.

- If you are using InstallShield from within Visual Studio: On InstallShield toolbar, click the **Visual Studio Deployment Project Import Wizard** button.

This wizard consists of the following panels:

- Welcome
- Project File
- Options
- Summary

Welcome Panel

The Visual Studio Deployment Project Import Wizard enables you to import a Visual Studio setup or merge module project (.vdproj) into an InstallShield project (.ise). You can reuse this wizard if you want to import multiple Visual Studio projects into your InstallShield project.

Important ▪ *If the Visual Studio setup or merge module project that you want to import into an InstallShield project contains one or more project outputs, the InstallShield project must be in the same Visual Studio solution that contains the Visual Studio setup or merge module project and all of its project dependencies.*

Click Next to begin using the wizard.

Project File Panel

The Project File panel is where you specify the Visual Studio project (.vdproj) that you want to import. The project can be a setup project or a merge module project.

Options Panel

The Visual Studio Deployment Project Import Wizard imports the project outputs, files, registry keys, file extensions, custom actions, target system searches, and prerequisites from your Visual Studio project into your InstallShield project. The Options panel is where you select which, if any, properties you also want InstallShield to import from the Visual Studio project into your InstallShield project.

Note ▪ *If you select the check box for an option, InstallShield overwrites the existing value in your InstallShield project with the value that is configured in the Visual Studio project. For example, if you select the Product Name*

check box, InstallShield overwrites the value of the Product Name setting of your InstallShield project with the value that is set in the Visual Studio project.

Table 9-17 ▪ Available Options for Importing

Option	Description
Product Name	If you want your InstallShield project to use the ProductName property that is configured in the Visual Studio project that you are importing, select this check box. The product name is configured in the General Information view of InstallShield. ***Project*** *▪ This option does not apply to merge module projects.*
Product Version	If you want your InstallShield project to use the value of the ProductVersion property that is configured in the Visual Studio project that you are importing, select this check box. The product version is configured in the General Information view of InstallShield.
INSTALLDIR	If you want your InstallShield project to use the value of the DefaultLocation property that is configured for the application folder in the Visual Studio project that you are importing, select this check box. If you select this check box, InstallShield updates the value of the INSTALLDIR setting in the General Information view with the path that is configured in the Visual Studio project. ***Note*** *▪ Visual Studio lets you specify a directory path that contains multiple formatted properties, such as [ProgramFilesFolder][Manufacturer]\[ProductName], for the application folder. Visual Studio projects use a directory custom action to resolve the path at run time. However, InstallShield does not support this type of directory path. Therefore, InstallShield resolves the path during the conversion process and uses the* `INSTALLDIR` *property for the path.* ***Edition*** *▪ The Premier and Professional editions of InstallShield have 64-bit support, but the Express edition does not.* *If the DefaultLocation property in your Visual Studio project uses a 64-bit location such as* ***[ProgramFiles64Folder]****, and if you select the INSTALLDIR check box on this panel of the Visual Studio Deployment Project Import Wizard, InstallShield Express Edition uses the 32-bit equivalent of the folder (for example,* ***[ProgramFilesFolder]****). The Premier and Professional editions use the 64-bit location.*

Table 9-17 ▪ Available Options for Importing (cont.)

Option	Description
Add or Remove Programs Properties	If you want your InstallShield project to use the Add or Remove Program properties (AddRemoveProgramsIcon, Manufacturer, Description, ManufacturerUrl, Author, SupportUrl, and SupportPhone) that are configured in the Visual Studio project that you are importing, select this check box. If you select this check box, InstallShield updates the values of the following settings in the General Information view with the values that are configured in the Visual Studio project: ● Display Icon (In the Visual Studio project, this is the AddRemoveProgramsIcon property.) ● Publisher (In the Visual Studio project, this is the Manufacturer property.) ● Add or Remove Programs Comments (In the Visual Studio project, this is the Description property.) ● Publisher/Product URL (In the Visual Studio project, this is the ManufacturerUrl property.) ● Support Contact (In the Visual Studio project, this is the Author property.) ● Support URL ● Support Phone Number ***Project ▪*** *This option does not apply to merge module projects.*

Table 9-17 ▪ Available Options for Importing (cont.)

Option	Description
Summary Information Stream Properties	If you want your InstallShield project to use the Summary Information Stream properties (Title, Subject, Keywords, and TargetPlatform) that are configured in the Visual Studio project that you are importing, select this check box. If you select this check box, InstallShield updates the values of the following settings in the General Information view with the values that are configured in the Visual Studio project: ● Title ● Subject ● Keywords ● Processor type in the Template Summary setting (In the Visual Studio project, this is the TargetPlatform property.) ***Edition ▪*** *Support for importing the TargetPlatform value is available in the Premier and Professional editions of InstallShield.* ***Project ▪*** *This option does not apply to merge module projects.*
Product Code	If you want your InstallShield project to use the value of the ProductCode property that is configured in the Visual Studio project that you are importing, select this check box. The product code is configured in the General Information view of InstallShield. ***Project ▪*** *This option does not apply to merge module projects.*
Upgrade Code	If you want your InstallShield project to use the value of the UpgradeCode property that is configured in the Visual Studio project that you are importing, select this check box. The upgrade code is configured in the General Information view of InstallShield. ***Project ▪*** *This option does not apply to merge module projects.*

Table 9-17 ▪ Available Options for Importing (cont.)

Option	Description
All Users	If you want your InstallShield project to use the value of the InstallAllUsers property that is configured in the Visual Studio project that you are importing, select this check box. If you select this check box, InstallShield updates the value of the ALLUSERS setting in the General Information view based on the value that is set in the InstallAllUsers property in the Visual Studio project. ***Project*** *▪ This option does not apply to merge module projects.*
Project Language	If you want your InstallShield project to use the language that is selected in the Localization property of the Visual Studio project that you are importing, select this check box. The language is configured in the Setup Language setting of the General Information view of InstallShield. Note that if you select this check box, and if the language of your Visual Studio project does not match the language in your InstallShield project, InstallShield replaces the existing string entry values in your project with default string entry values for the language of your Visual Studio project. For example, if you select this check box, if the language of your InstallShield project is Spanish, and if the language of your Visual Studio project is German, InstallShield replaces the Spanish run-time strings in your project with the default German translations. Thus, if you edit a string entry value by revising a setting such as the Publisher setting in the General Information view, and then you specify to import the language of a Visual Studio project, InstallShield overwrites the value of the Publisher setting—as well as values for other settings—with the default German string entry values. ***Project*** *▪ This option does not apply to merge module projects.*

Summary Panel

The Summary panel lets you review the settings that you specified in the Visual Studio Deployment Project Import Wizard. To change any of the settings, click the Back button until you reach the appropriate panel. To import the project, click the Finish button.

Web Deployment Wizard

The Web Deployment Wizard enables you to build a setup for Web deployment. Using this wizard, you can create a Web page automatically to link to your setup, specify whether you want to include the Windows Installer engine, and provide digital signature and security information. The following panels are included in this wizard:

- Welcome
- Link Type
- Windows Installer Engine Options
- Windows Installer Location
- Advanced Settings
- Digital Signature and Security for Targeting Internet Explorer
- Summary

Welcome Panel

The Web Deployment Wizard enables you to build a setup for Web deployment. Using this wizard, you can create a Web page automatically to link to your setup, specify whether you want to include the Windows Installer engine, and provide digital signature and security information.

Link Type Panel

The Link Type panel lets you select the type of hyperlink to your release that you want on the Web page. The link can be either one that relies on One-Click Install technology to automatically begin the installation, or one that displays a dialog from which the end user is prompted to run or save the installation.

Select Yes to create a One-Click Install link; select No to display the standard Save/Run dialog when the end user clicks the link.

Windows Installer Engine Options Panel

The Windows Installer Engine Options panel lets you specify whether you want to include the Windows Installer installation in your installation. You can configure the following options:

Table 9-18 ▪ Windows Installer Engine Options Panel Settings

Setting	Description
Include Windows Installer 3.1 engine—requires Windows 2000 SP3 or higher	If you want to include the Windows Installer 3.1 engine in your installation, select this check box.

Table 9-18 ▪ Windows Installer Engine Options Panel Settings (cont.)

Setting	Description
Suppress launcher warning	If the operating system has a version of the Windows Installer service that cannot be updated by the installation, you can suppress the warning that is displayed to the end user.
Delay Windows Installer reboot	Select this check box to delay any reboot that may be required by the Windows Installer engine installation until your installation has completed. Clear this check box to allow the system to reboot immediately after installing or updating the Windows Installer engine, if necessary.

Windows Installer Location Panel

If you have selected to include the Windows Installer engine in your setup, you have two options as to where the installation will look for the setup for the engine:

Table 9-19 ▪ Windows Installer Location Panel Settings

Setting	Description
Download engine from the Web as specified below	If you want your installation to download the Windows Installer engine redistributables (if necessary) from the URL that is specified, select this option. The default URL (http://www.installengine.com/Msiengine30) is maintained by Revenera for your convenience. The advantage in selecting this option is that it reduces the size of your installation; however, the end user must have an Internet connection if the Windows Installer engine is needed, and it may take time to download the engine's installation.
Extract engine from Setup.exe	If you want to compress the selected Windows Installer engine redistributable file or files into Setup.exe, to be extracted at run time, if necessary, select this option. Although selecting this option increases the size of the Setup.exe, the result may be a quicker installation, since the engine does not need to be downloaded. In addition, this option does not require an Internet connection.

Advanced Settings Panel

There are two additional options available in the Advanced Settings panel:

Table 9-20 ▪ Advanced Settings Panel Settings

Setting	Description
Optimize for media size	If this option is selected, the size of the Setup.exe will be reduced. However, this takes more time during the build process.

Table 9-20 ▪ Advanced Settings Panel Settings (cont.)

Setting	Description
Copy media to target machine	If you want the .msi file and other installation files to be cached on the target system, select this check box and specify the cache location. If you do not want the files to be cached, clear this check box. Caching the .msi file and other files on the target system is useful for application maintenance and repair.
Password protect Setup.exe with the following password	If you want to password-protect your installation to require end users to enter a valid password when installing your project, select this check box, and enter the valid password.

Digital Signature and Security for Targeting Internet Explorer Panel

The Digital Signature and Security panel is where you specify the digital signature information—including the digital certificate files that a certification authority grated to you—that InstallShield should use to sign your files.

Table 9-21 ▪ Digital Signature and Security for Targeting Internet Explorer Panel Settings

Setting	Description
Sign Media for Internet Explorer	If you want to digitally sign your installation package, select this check box, and then complete the other settings on this panel.
URL	Type a fully qualified URL—for example, **http://www.mydomain.com**. This URL is used in your digital signature to link to a site that you would like end users to visit to learn more about your product, organization, or company.

Table 9-21 ▪ Digital Signature and Security for Targeting Internet Explorer Panel Settings (cont.)

Setting	Description
Digital certificate information	To specify the digital certificate that you want to use to sign your release, click the ellipsis button (...) next to this setting. The Certificate Selection dialog box opens, enabling you to specify either the location of the .pfx file, certificate file (EV exported .cer file), or information about the certificate store that contains the certificate. InstallShield provides an option to encrypt and store an EV token password in the project file using the public key certificate (.cer) file. The .cer file is generally created by exporting a public key from the EV Authentication Client tools associated with a USB eToken provider (for example, SafeNet Authentication Client). InstallShield displays additional options to configure the Extended Validation (EV) certificate properties if the .cer file is specified. For more details, see ● Configuring Extended Validation (EV) Certificate information in InstallShield ● Certificate Selection Dialog Box After specifying the .pfx file or choosing the certificate from test store, the below will be displayed: ● **Certificate Thumbprint**—This read-only setting displays the certificate thumbprint. ● **Issued By**—This read-only setting displays the certificate issuer information. ● **Expiration Date**—This read-only setting displays the certificate's expiration date. After specifying the certificate file (EV exported .cer file), the below settings will be displayed: ● **Cryptographic Provider**—This setting allows you to specify the cryptographic service provider (CSP). ● **Container Name**—This setting allows you to specify the private key container name associated to the cryptographic service provider (CSP). ● **Token Password**—This setting allows you to specify the EV token password which is encrypted and stored in a project file. ***Note ▪*** *Note the following informations while utilizing the settings that appear after specifying the certificate file (EV exported .cer file):* ● *Both the Cryptographic Service Provider (CSP) name and Container name can be obtained from the Private Key Certificate properties of a user certificate in the EV Vendor Authentication Client tool.* ● *An EV certificate vendor determines an EV token password's expiration period and number of invalid password attempts before it is locked. Therefore, selecting this setting requires changing your password in specific intervals.* ● *If an EV token password is locked, unlocking/resetting the token password requires an administrator password.*

Table 9-21 ▪ Digital Signature and Security for Targeting Internet Explorer Panel Settings (cont.)

Setting	Description
Certificate password	If the .pfx file that you are using has a password, enter it. InstallShield encrypts the password and stores it in your project file (.ise). At build time, InstallShield uses the password to sign files with a .pfx file. If your certificate is protected by a password but you do not enter it in this setting, signing with a .pfx file fails. Note that if you configure your project to use a certificate that was imported with password protection into a store, Windows prompts for the password at build time when InstallShield is attempting to sign your project's files. The strong key protection that Windows uses does not permit InstallShield to provide the password to the cryptographic provider.

Summary Panel

The Summary panel displays the configuration settings you supplied in the previous panels for your Web Deployment. Review these settings to ensure accuracy. Depending on the option selected, clicking Finish either exits the Web Deployment wizard and builds the release, or only exits.

View Reference

The InstallShield installation development environment (IDE) is organized into views that incorporate various areas of functionality. The View Reference section describes each of those views in the InstallShield interface.

Organize Your Setup View

The first step in building anything is to lay a solid foundation. The base of your installation is formed by specifying application information through the General Information view, creating features in the Features view, specifying setup types in the Setup Types view, and preparing for update notification in the Update Notifications view. Each view under Organize Your Setup is described below:

Table 9-1 ▪ Views Under the Organize Your Setup View

View	Description
General Information	The General Information view contains basic information about your project, your company, and your product. When you create a new installation project, you must configure its General Information view settings. InstallShield creates new projects with default settings, but you should set your own values so that your project includes data that is specific to your needs.
Features	Features are the building blocks of your setup. They represent a distinct piece of your application—such as program files, help files, or clip art—to your end users. You can create features and subfeatures in the Features view.
Setup Types	Setup types offer different configurations of your product to your end users. These configurations can be useful if you distribute large features that are not required in order for the product to run.
Upgrade Paths	If you have distributed previous versions of your application and want the current version to update the end user's system if the previous version is installed, you can indicate upgrade information in the Upgrade Paths view.
Update Notifications	The Update Notifications view is where you enable FlexNet Connect for your original installation. ***Caution ▪*** *Enabling automatic update notifications in your project adds about 600 KB of files to your installation. These files must be distributed with your application in order for FlexNet Connect to work. If you cannot afford to include these files in your installation because of bandwidth limitations or other reasons, you can select No to disable automatic update notifications. However, FlexNet Connect cannot be used to deploy an update unless automatic notification is enabled in the original installation when you distribute it to your end users. Therefore, if you select No, you will not be able to later take advantage of the automatic update notification features.*

General Information View

The General Information view contains basic information about your setup, your company, and the application that you are installing. Some of the information you enter in this view is for your reference only, some is necessary to comply with Windows logo requirements, and the rest is for configuring basic installation settings.

When you create a new installation project, you must configure its General Information view settings. InstallShield creates new projects with default settings, but you should set your own values so that your project includes data that is specific to your needs.

The General Information view consists of the following elements:

- A row of buttons
- A grid of settings

The following table describes the buttons that are displayed in the General Information view.

Table 9-2 ▪ Controls in the General Information View

Name of Control	Icon	Description
Categorized		Sorts the settings according to categories.
Alphabetical		Sorts the settings alphabetically.

For descriptions of each of the settings in the General Information view, see General Information Settings.

General Information Settings

The General Information settings are organized into the following main categories:

- General
- Summary Information Stream
- Add or Remove Programs
- Software Identification Tag

General Settings

Use the General area of the General Information view to specify details such as product name and product version. The following settings are available in this area.

Table 9-3 ▪ General Settings

Setting	Description
Product Name	Enter the name of the product. For information on how the product name is used, see Specifying a Product Name.

Table 9-3 ▪ General Settings (cont.)

Setting	Description
Product Version	Enter the version number for your product. The version must contain only numbers. It is typically in the format *aaa.bbb.ccccc* or *aaa.bbb.ccccc.ddddd*, where *aaa* represents the major version number, *bbb* represents the minor version number, *ccccc* represents the build number, and *ddddd* represents the revision number. The maximum value for the *aaa* and *bbb* portions is 255. The maximum value for *ccccc* and *ddddd* is 65,535. Note that although you can include the fourth field (*ddddd*), the installation does not use this part of the product version to distinguish between different product versions. To learn more, see Specifying the Product Version. If your release includes a `Setup.exe` file, the product version that you specify is displayed on the Properties dialog box for `Setup.exe`. For more information, see Customizing File Properties for the Setup Launcher.
Product Code	Enter a GUID that uniquely identifies this product. To have InstallShield generate a different GUID for you, click the **Generate a new GUID** button ({...}) in this setting. Since this code uniquely identifies your product, changing the product code after you have already distributed your release is not recommended. For more information, see Setting the Product Code.
Upgrade Code	Enter a GUID that uniquely identifies this product. To have InstallShield generate a different GUID for you, click the **Generate a new GUID** button ({...}) in this setting. The product GUID is used to associate uninstallation or maintenance with the original installation. A new GUID is automatically generated for each new project that you create, including copies of existing projects. Once you have changed a project's product GUID, its previous GUID cannot be recovered. For these reasons, changing a project's product GUID is typically not necessary and should be approached with caution. For more information, see Setting the Upgrade Code.

Table 9-3 ▪ General Settings (cont.)

Setting	Description
INSTALLDIR	Specify the value of the Windows Installer property `INSTALLDIR`, which indicates the destination directory where most of the files of the product will be installed at run time. The default value is as follows: `[ProgramFilesFolder]`*`My Company Name`*`\`*`My Product Name`* Instead of hard-coding a path, you can enter a directory property as part of the path. To select a directory property, click the ellipsis button (...) in this setting. This enables you to select the appropriate directory from a list, or to create a new directory within a predefined directory. Separate further levels of subdirectories with a backslash—for example, **`[ProgramFilesFolder]MyApp\Bin`**. ***Windows Logo ▪*** *To comply with the Windows logo program, your product's default destination should be a subfolder of the Program Files folder (***`[ProgramFilesFolder]`***), which can vary depending on the system's locale and user settings.* For more information, see Setting the Default Product Destination Folder (INSTALLDIR).

Table 9-3 ▪ General Settings (cont.)

Setting	Description
Locked-Down Permissions	Select the type of permissions that you want to use for securing files, folders, and registry keys for end users who run your product in a locked-down environment. Available options are: • **Custom InstallShield handling**—InstallShield uses a custom `ISLockPermissions` table and adds custom actions to your project to set permissions on the target system. This option is the default value. • **Traditional Windows Installer handling**—InstallShield uses the `LockPermissions` table in the .msi database to store permission information for your product. It is often more advantageous to use the custom InstallShield handling than the traditional Windows Installer handling. For example: • The custom option includes support for many well-known security identifiers (SIDs) that are not supported by the traditional option. • The custom option supports the use of localized user names for many well-known SIDs, unlike the traditional option. With the traditional option, if you try to use a localized name to set permissions on a non-English system, the installation may fail. • The custom option lets you specify that you want to deny a user or group from having the permissions that you are specifying. The traditional handling does not allow you to do this. That is, with the traditional handling, you can only set specific permissions; you cannot deny permissions. This is a project-wide setting that affects all new permissions that you set for files, folders, and registry keys in your project. If you have already configured some permissions in your project and then you change the value of this setting, InstallShield lets you specify whether you want to use the alternate handling method for those already-existing permissions. For more information about configuring this setting, see Securing Files, Folders, and Registry Keys in a Locked-Down Environment.

Table 9-3 ▪ General Settings (cont.)

Setting	Description

DATABASEDIR	Specify the default destination folder for all of your application's database files.
	
	Note ▪ *When you are specifying a destination folder for your application's database files, you cannot include a space after the closing square bracket (]), or before or after a backslash (\). For example, the following paths are not valid:*
	`[ProgramFilesFolder] \MyApp\Bin`
	`[ProgramFilesFolder]MyApp\ Bin`
	It is important to note when using an installer folder property such as `DATABASEDIR` that you are specifying a default value. An end user could change this value by setting a property when launching `Msiexec.exe` at the command line or by selecting a new destination folder for a feature in the Database Folder dialog.

Default Font	To select the font that you want your installation to use in its user interface, click the ellipsis button (...) in this setting. If you do not select a font, the installer uses the default system font.

Table 9-3 ▪ General Settings (cont.)

Setting	Description
ALLUSERS	Specify how the Windows Installer property ALLUSERS should be set in your project. The ALLUSERS property specifies whether Windows Installer should attempt to perform a per-machine or per-user installation. Valid options are: • **ALLUSERS="" (Per-user installation)**—Windows Installer performs a per-user installation, and the configuration information is stored in the user's personal profile. • **ALLUSERS=1 (Per-machine installation)**—If the value of ALLUSERS is set to 1, Windows Installer attempts a per-machine installation. For per-machine installations, configuration information such as shortcuts and registry entries are stored in the All Users profile. On Windows Vista and later systems, if User Account Control is enabled and the user does not have administrative privileges, the user must be able to provide administrative credentials in order to install the product. On other systems, if the user does not have administrative privileges, the installation displays an error message and exits. • **ALLUSERS=2 (Per-user or per-machine installation)**—Windows Installer attempts to perform a per-machine installation on Windows Vista and later systems. On earlier platforms, if the user has administrative privileges, Windows Installer attempts to perform a per-machine installation; otherwise, Windows Installer performs a per-user installation. The default value for all new projects is ALLUSERS=1 (Per-machine installation). ***Note ▪*** *The value that you specify here for* ALLUSERS *could be overwritten at run time. For more information, see Per-User vs. Per-Machine Installations.*

Table 9-3 ▪ General Settings (cont.)

Setting	Description
Create MSI Logs	To specify whether Windows Installer 4.0 or later should log your installation, click the ellipsis button (...) in this setting, which launches the Logging Options for Windows Installer 4.0 and Later dialog box. This dialog box is where you specify whether Windows Installer should log your installation. You can also use this dialog box to customize the types of messages that are logged. There are three possible values for this setting: ● **No**—Installations are not logged. This is the default value. ● **Yes**—InstallShield populates the `MsiLogging` property with the default value of **voicewarmupx**. ● **Custom**—InstallShield populates the `MsiLogging` property with the value that you specified on the Logging Options for Windows Installer 4 and Later dialog box. If the value of this setting is Yes or Custom, and if the installation is run with Windows Installer 4.0 or later, as well as Windows Vista or later or Windows Server 2008 or later, the following occurs: ● The installer creates a log file according to the appropriate logging mode: either **voicewarmupx** (if the Create MSI Logs value is Yes) or whatever custom value you specified on the Logging Options for Windows Installer 4.0 and Later dialog box. ● The installer populates the `MsiLogFileLocation` property with the log file's path. ● A **Show the Windows Installer log** check box is added to the SetupCompleteSuccess, SetupCompleteError, and SetupInterrupted dialogs. If the end user selects that check box and then clicks Finish, the log file is opened in a text file viewer or editor. The Create MSI Logs setting applies to installations that are run with Windows Installer 4.0 or later on Windows Vista and later systems or Windows Server 2008 and later systems. The **Show the Windows Installer log** check box is not visible in run-time dialogs that are displayed on earlier systems that are running earlier versions of Windows Installer. For more information, including details on how to customize the types of messages that are logged, see Specifying Whether Windows Installer Installations Should Be Logged.

Table 9-3 ▪ General Settings (cont.)

Setting	Description
Fast Install	If you want to reduce the time that is required to install a large Windows Installer package, consider selecting the check boxes for one or more of the following options: • No system restore point is saved for this installation • Perform only File Costing and skip checking other costs • Reduce the frequency of progress messages This setting configures the Windows Installer property `MSIFASTINSTALL`, which can be set at the command line. Windows Installer 5 includes support for this setting. Earlier versions of Windows Installer ignore it.

Summary Information Stream Settings

Windows Installer databases are implemented as COM structured storage, and COM structured storage files usually contain a Summary Information Stream. The Summary Information Stream contains information about your company and the product that is being installed.

The following settings are available in the Summary Information Stream area in the General Information view.

Table 9-4 ▪ Summary Information Stream Settings

Setting	Description
Title	Specify the type of database that you are creating. For a product installation, the default value is **Installation Database**, which is the recommended value. The value that you enter is used on the Summary tab of the Properties dialog box that is displayed if you right-click the Windows Installer database and then click Properties.
Subject	Enter the name of the product. The value that you enter is used on the Summary tab of the Properties dialog box that is displayed if you right-click the Windows Installer database and then click Properties.
Author	Specify your company name. The value that you enter is used on the Summary tab of the Properties dialog box that is displayed if you right-click the Windows Installer database and then click Properties.
Keywords	Specify keywords that describe the Windows Installer database for your product. The value that you enter is used on the Summary tab of the Properties dialog box that is displayed if you right-click the Windows Installer database and then click Properties.

Table 9-4 ▪ Summary Information Stream Settings (cont.)

Setting	Description
Summary Information Stream Comments	Enter any comments about your product. A typical value for this setting is as follows: `This installer database contains the logic and data required to install MyProduct.` The value that you enter is used on the Summary tab of the Properties dialog box that is displayed if you right-click the Windows Installer database and then click Properties.
Schema	This setting lets you specify the integer that identifies the minimum Windows Installer version that is required for your installation package. For a minimum of Windows Installer 2.0, enter **`200`**. For a minimum of Windows Installer 3.0, enter **`300`**. For a minimum of Windows Installer 3.1, enter **`301`**. For a minimum of Windows Installer 4.5, enter **`405`**. If the end user's system has a Windows Installer version earlier than the minimum requirement that you specify for the Schema setting—for example, if you specify a schema value of 405 because your installation uses Windows Installer 4.5 features, but an end user has Windows Installer 3.1—the installation displays an error message and exits. The value that you enter for the Schema setting is used for the Page Count Summary property of your Windows Installer database.
Require Administrative Privileges	Specify whether your installation requires administrative privileges. The default is Yes. If you set this to No, InstallShield sets bit 3 in the Word Count Summary property to indicate that elevated privileges are not required to install the .msi package. Note that if you select No but your .msi package tries to perform a task for which it does not have adequate privileges, Windows Installer may display a run-time error. This setting applies to installations that are run with Windows Installer 4.0 or later on Windows Vista and later systems or Windows Server 2008 and later systems. Earlier versions of Windows Installer and Windows ignore this setting. Note that an end user's installation experience is more secure when installations are run with only the permissions that they need. Unless an application is designed to be run only by system administrators, it should be run with the least privilege.

Add or Remove Programs Settings

Add or Remove Programs (which is called Programs in the latest versions of Windows) in the Control Panel provides end users with technical support links and telephone numbers, product update information, and information about a product's manufacturer. Depending on how the installation is configured, the end user may have the option of removing, repairing, or changing the installation with the click of a button. You can specify this information in your project by configuring the Add or Remove Programs settings in the General Information view.

Table 9-5 ▪ Add or Remove Programs Settings

Setting	Description
Show Add or Remove Programs Entry	Indicate whether you want to show your product's entry in Add or Remove Programs in the Control Panel. Available options are: • **Yes**—Your product's entry is displayed on the target system in Add or Remove Programs. This is the default value. • **No**—Your product's entry is not displayed on the target system in Add or Remove Programs. The end user cannot use Add or Remove Programs to remove the product, perform maintenance, or view support information. If you select this option, InstallShield disables the other Add or Remove Programs settings.
Display Icon	Enter the path on your development system to the icon file (.ico or .exe) that you want to be used for your product's entry in Add or Remove Programs. Instead of manually typing the path and file name, you can click the ellipsis button (...) in this setting to browse to the file.
Disable Change Button	Specify whether you want to disable the Change button for your product's entry in Add or Remove Programs. The Change button enables end users to change installation options after the product has been installed. End users can remove or add features as needed.
Disable Remove Button	Specify whether you want to disable the Remove button for your product's entry in Add or Remove Programs. The Remove button enables end users to remove the product by clicking one button, which runs your uninstaller with a reduced user interface. If the end user clicks the Remove button to remove your product, actions in the User Interface sequence of the project are executed.
Disable Repair Button	Specify whether you want to disable the Repair button for your product's entry in Add or Remove Programs. The Repair button enables end users to run the Windows Installer repair option if any files have been deleted or corrupted.

Table 9-5 ▪ Add or Remove Programs Settings (cont.)

Setting	Description
Publisher	Specify the name of the company that created the product. This information is displayed for your product's entry in Add or Remove Programs. The value that you enter is stored in the Windows Installer `Manufacturer` property. ***Windows Logo ▪*** *If you want to meet the requirements of the Windows logo program, you must specify the publisher.*
Publisher/Product URL	Enter a general URL for your company or product—for example, **http://www.installshield.com**. On some versions of Windows, the publisher name on the Support Info dialog box is a hyperlink to this URL. The Support Info dialog box is displayed when an end user clicks the support information hyperlink for your product's entry in Add or Remove Programs.
Support Contact	Enter the name of the person or department that end users should contact for technical support. On some versions of Windows, this information is displayed on the Support Info dialog box for your product's entry in Add or Remove Programs.
Support URL	Enter the URL that you would like end users to visit for technical support information for your product. This URL is displayed for your product's entry in Add or Remove Programs. ***Windows Logo ▪*** *If you want to meet the requirements of the Windows logo program, you must enter a valid URL.*
Support Phone Number	Enter the technical support phone number for your product. On some versions of Windows, this information is displayed on the Support Info dialog box for your product's entry in Add or Remove Programs.
Read Me	Enter the name or path of the Readme file for your product. As an alternative, you can link to a Readme file located on the Internet by specifying a valid URL. On some versions of Windows, this information is displayed on the Support Info dialog box for your product's entry in Add or Remove Programs. ***Note ▪*** *You do not need to use a backslash (\) between a directory identifier that you select from the list and the file or folder name. For example,* **[INSTALLDIR]MyFolder\Readme.txt** *is a valid path format.*

Table 9-5 ▪ Add or Remove Programs Settings (cont.)

Setting	Description
Product Update URL	Specify a URL where end users can get information about product updates or download the latest version. On some versions of Windows, this information is displayed on the Support Info dialog box for your product's entry in Add or Remove Programs.
Add or Remove Programs Comments	Enter comments about your product. This information is displayed for your product's entry in Add or Remove Programs.

Software Identification Tag Settings

Use the Software Identification Tag area of the General Information view to specify whether you want to include an ISO/IEC 19770-2 software identification tag in your installation. If a tag is included, this area also lets you specify the identification information that is not already specified in other areas of the General Information view.

For more information, see Including a Software Identification Tag for Your Product.

Table 9-6 ▪ Software Identification Tag Settings

Setting	Description
Use Software Identification Tag	Specify whether you want to include an ISO/IEC 19770-2 software identification tag in your installation. If you select Yes, use the other tag-related settings in this area of the General Information view to specify the identification information that is not already specified in other areas of the General Information view.
Require Software Entitlement	Specify whether you want to require your product to have a corresponding software entitlement in order for software reconciliation to be considered successful. In general, if the software must be purchased, Yes should be selected for this setting; if the software is free, No should be selected for this setting.
Unique ID	Enter a unique ID that identifies the specific version of this specific product. To have InstallShield generate a different GUID for you, click the **Generate a new GUID** button (...) in this setting. Note that InstallShield uses the value that you enter as part of the name of the tag file (*TagCreatorID_UniqueID*.swidtag). Therefore, the ID that you enter must not contain any characters that are invalid for file names.
Tag Creator	Enter the name of the organization that created the tag.

Table 9-6 ▪ Software Identification Tag Settings (cont.)

Setting	Description
Tag Creator ID	Enter the registration ID of the organization that created the tag. This ID helps to differentiate between different legal organizations that have the same creator name but are in different countries. The convention for the registration ID is as follows: **regid.YYYY-MM.ReversedDomainName,division** The registration ID consists of the following parts: ● **regid.**—The string *regid* indicates that the XML portion is a registration ID for a software identification tag. A period (.) must be included after this string. ● **YYYY-MM.**—This part of the registration ID identifies the first full month (MM) and the year (YYYY) in which the domain name was owned by the tag creator. For example, if you are creating the tag and you purchased the domain name February 15, 1999, you would use **1999-03** in this part of the registration ID, since the first full month the domain name was owned was March (03), and the year was 1999. The year and month must be separated by a dash. ● **ReversedDomainName**—This part identifies the reversed domain name of the organization that is creating the software identification tag. For example, for the revenera.com domain name, the reversed domain name is: **com.revenera** ● **,division**—This optional part starts with a comma (,), and is followed by an additional string. You can enter a string that helps to distinguish between different divisions or areas of the organization. If you do not want to use this optional distinguishing part of the registration ID, do not include the comma or an additional string in your entry. Note that InstallShield uses the value that you enter as part of the name of the tag file (*TagCreatorID_UniqueID*.swidtag). Therefore, the ID that you enter must not contain any characters that are invalid for file names.
Software Creator	Enter the name of the organization that created the software. This setting is optional. If you leave this setting blank, InstallShield uses the value of the Tag Creator setting for the name of the software creator.

Table 9-6 ▪ Software Identification Tag Settings (cont.)

Setting	Description
Software Creator ID	Enter the registration ID of the organization that created the software. This ID helps to differentiate between different legal organizations that have the same creator name but are in different countries. This setting is optional. If you leave this setting blank, InstallShield uses the value of the Tag Creator ID setting for the software creator ID. The convention for the registration ID is as follows: **regid.YYYY-MM.ReversedDomainName,division** The registration ID consists of the following parts: ● **regid.**—The string *regid* indicates that the XML portion is a registration ID for a software identification tag. A period (.) must be included after this string. ● **YYYY-MM.**—This part of the registration ID identifies the first full month (MM) and the year (YYYY) in which the domain name was owned by the tag creator. For example, if you are creating the tag and you purchased the domain name February 15, 1999, you would use **1999-03** in this part of the registration ID, since the first full month the domain name was owned was March (03), and the year was 1999. The year and month must be separated by a dash. ● **ReversedDomainName**—This part identifies the reversed domain name of the organization that is creating the software identification tag. For example, for the revenera.com domain name, the reversed domain name is: **com.revenera** ● **,division**—This optional part starts with a comma (,), and is followed by an additional string. You can enter a string that helps to distinguish between different divisions or areas of the organization. If you do not want to use this optional distinguishing part of the registration ID, do not include the comma or an additional string in your entry.
Software Licensor	Enter the name of the organization that owns the copyright for the software. This setting is optional. If you leave this setting blank, InstallShield uses the value of the Tag Creator setting for the name of the software licensor.

Table 9-6 ▪ Software Identification Tag Settings (cont.)

Setting	Description
Software Licensor ID	Enter the registration ID of the organization that owns the copyright for the software. This ID helps to differentiate between different legal organizations that have the same licensor name but are in different countries. This setting is optional. If you leave this setting blank, InstallShield uses the value of the Tag Creator ID setting for the software licensor ID. The convention for the registration ID is as follows: **regid.YYYY-MM.ReversedDomainName,division** The registration ID consists of the following parts: ● **regid.**—The string *regid* indicates that the XML portion is a registration ID for a software identification tag. A period (.) must be included after this string. ● **YYYY-MM.**—This part of the registration ID identifies the first full month (MM) and the year (YYYY) in which the domain name was owned by the tag creator. For example, if you are creating the tag and you purchased the domain name February 15, 1999, you would use **1999-03** in this part of the registration ID, since the first full month the domain name was owned was March (03), and the year was 1999. The year and month must be separated by a dash. ● **ReversedDomainName**—This part identifies the reversed domain name of the organization that is creating the software identification tag. For example, for the revenera.com domain name, the reversed domain name is: **com.revenera** ● **,division**—This optional part starts with a comma (,), and is followed by an additional string. You can enter a string that helps to distinguish between different divisions or areas of the organization. If you do not want to use this optional distinguishing part of the registration ID, do not include the comma or an additional string in your entry.

Features View

Features and subfeatures give you almost limitless flexibility in packaging your application and related accessories for setup. You can create and configure Features from the Features view.

A feature is a building-block of an application from the end user's perspective. That is, they represent a specific capability of your product, such as its help files or a part of a product suite that can be installed or uninstalled based on the end user's selections. Your entire application should be divided into features that perform a specific purpose.

To make it easier to lay out a complex application, InstallShield allows you to create both features and subfeatures. Subfeatures are further divisions of a feature. Since features should be self-contained elements of an application or application suite that a user can selectively install, it might make the most sense for you to organize portions of your application as subfeatures of a "parent" feature. Assuming that the features are all visible, your end user can then select which portions of a feature to install in the Custom Setup dialog.

The default feature Always Install cannot be renamed or removed from your project, nor can you add any subfeatures to it. This feature should contain files that must be installed as part of your setup.

Although you can create up to 15 levels of subfeatures, you should keep the design as simple as possible for organizational purposes.

Feature Settings

Feature settings are as follows.

Table 9-7 ▪ Feature Settings

Setting	Description
Description	Enter a brief description of the feature. A feature's description is displayed in the Custom Setup dialog when the end user clicks a feature or subfeature
Remote Installation	Indicate the default install state for the feature. Valid options are: • **Favor Source**—The files in this feature are set to be run directly from the source medium, such as a CD-ROM or a network location, by default. • **Favor Local**—The files in this feature are set to be installed on the target system by default. • **Favor Parent**—The subfeature has the same default state as its parent feature. This option is available only if the feature is a subfeature. For more information, see Setting a Feature's Remote Installation Setting.
Visible	By configuring a feature's Visible setting, you specify how you want it presented to the end user in the Custom Setup dialog. Select one of the following options: • **Visible and Collapsed**—The feature is displayed in the Custom Setup dialog with its subfeatures collapsed by default. • **Visible and Expanded**—The feature is displayed in the Custom Setup dialog with its subfeatures expanded by default. • **Not Visible**—The feature is not displayed to the end user in the Custom Setup dialog. This setting does not have any direct impact on whether a feature is installed. A feature is not automatically installed if it is not visible—it just cannot be deselected if it would otherwise be installed, or selected if it should not be installed. For more information, see Displaying Features to End Users.

Table 9-7 ▪ Feature Settings (cont.)

Setting	Description
Advertised	Select the appropriate advertisement option for this feature. Available options are: • **Allow Advertise**—End users have the ability to select the advertisement option for this feature in the Custom Setup dialog. Although advertisement is allowed, it is not the default option when the installation is run. • **Favor Advertise**—The feature is advertised by default. End users can change the advertisement option for a feature in the Custom Setup dialog. • **Disallow Advertise**—Select this option if you do not wish to allow advertising for this feature. Your end users will not be able to choose to have a feature advertised in the Custom Setup dialog. • **Disallow Advertise if not supported**—Advertisement works only on systems with Internet Explorer 4.01 or later. If the target system does not meet this criterion, advertising is not allowed. If the target system can support advertisement, advertising is allowed. For more information, see Advertising Features.
Required	Indicate whether the feature is required on the target system; if the feature is required, end users cannot deselect it in the Custom Setup dialog.
Condition	This setting indicates whether any conditions are set for this feature. To specify one or more conditions, click the ellipsis button (...) in this setting. For more information, see Setting Feature Conditions. When you add a condition, InstallShield adds a new row to the grid of feature settings for that condition. To add a condition, or to edit or delete one of the existing conditions, click the ellipsis button (...) in the Condition setting.
Comments	Enter comments to help you track the changes you make to a feature, or for any future reference. The feature's comments are stored only in the project file and are not used in the installation at any time.
REG File to Merge at Build	If you want to merge a .reg file with the feature's registry entries at build time, enter the complete path for the .reg file. As an alternative, you can click the ellipsis button (...) in this setting to browse to the .reg file.
Key Name	This read-only setting shows the Windows Installer key that is used to represent your feature. You can use this key in custom actions to access the feature.

Setup Types View

In the Setup Types view, you can configure the setup types that appear during the installation, the setup type names, and the features that are associated with each setup type. Following is an explanation of how these setup types are typically defined:

Table 9-8 ▪ Setup Types

Setup Type	Description
Typical	This setup type normally includes most, if not all, of a program's features. For example, if your setup includes multimedia tutorials, they would be included as part of the Typical setup.
Minimal	This setup type usually includes those features absolutely necessary in order for your application to run. This setup type is designed for those people who have limited disk space, such as notebook computers.
Custom	This setup type allows your users to select which features they want to install. Of course, those features that are required should be marked as such, thereby ensuring they are always installed. However, features such as the online help may not need to be installed. In these cases, the end user can select which of the unnecessary features to install.

When you select a setup type in the Setup Types view, the following setting is available:

Table 9-9 ▪ Setup Types

Setting	Description

Description	Enter a description for this setup type. The description is displayed to end users on the Setup Type dialog.
	
	To enter a multi-line description, use the **\n** escape sequence. For example, if you type **`InstallShield\nExpress`**, the description in the Setup Types dialog appears as:
	
	`InstallShield`
	
	`Express`

Upgrade Paths View

In the Upgrade Paths view, you can indicate information specific to how you want your product to be updated. For example, you can provide the upgrade code (product GUID), minimum and maximum versions, language identifiers, and other information for any previous versions of your product for which you want the current version to update.

You can also use the Upgrade Paths view to work around an issue that causes an assembly to be removed from the global assembly cache (GAC) of the target system when a major upgrade is applied. For more information, see Preventing the Removal of Assemblies from the Global Assembly Cache During Upgrades.

Upgrade Path Settings

By specifying upgrade path settings, you indicate how you want to affect an end user's system if a previously released version of your product that uses Windows Installer is installed—and that installed version has a different product code and a different version number than your current release. Each upgrade entry setting is described below:

Table 9-10 ▪ Upgrade Path Settings

Setting	Description
Upgrade Code	This setting contains as its value the upgrade code GUID of the product version that you want this version of your application to replace. When the installation of your current version begins, the Windows Installer engine searches the target system for the upgrade code specified. Upon finding a matching upgrade code—along with other matching upgrade properties—the Windows Installer upgrades the target system by installing the new version. If you added an upgrade entry to the Upgrade Paths tree by browsing for and selecting an .msi file on your system, the upgrade code of that .msi file automatically populates the Upgrade Code setting. If the .msi file for the product version that you want to upgrade is not on your system, you can type the upgrade code in this setting. You can also use the ellipsis (...) button to navigate to an .msi file (or a `Setup.exe` file that contains an .msi file) on your system and populate the Upgrade Code setting. ***Caution ▪*** *The upgrade code must be the same as that of the previous version of your product that was installed using Windows Installer technology.*

Table 9-10 ▪ Upgrade Path Settings (cont.)

Setting	Description
Min Version	The Min Version setting further defines the Windows Installer's search for installed versions to upgrade. The previously installed version's upgrade code must first match the value specified in the Upgrade Code setting. Once a matching upgrade code match is detected, the search can be narrowed by the minimum and/or maximum version specified and whether the versions are included or excluded (indicated in the Include Min Version and Include Max Version settings). This setting contains the minimum version number of the product that you want your current version to update. The setting can be populated in one of the following ways: ● If you added an upgrade entry to the Upgrade Paths tree by browsing for and selecting an .msi file or a `Setup.exe` file on your system, InstallShield uses the version of the .msi file in the Min Version setting. ● You can also type the minimum version in this setting in the format 0.00.0000 or 00.00.0000 (where the zeros represent the version number). ***Caution ▪*** *You must specify a value for at least one of the version settings: Min Version, Max Version, or both.*
Include Min Version	Specify whether you want to include the minimum version—specified in the Min Version setting—in the upgrade. If you want the Windows Installer to include the minimum version in the search for previously installed versions of your product on the target system, select Yes.

Table 9-10 ▪ Upgrade Path Settings (cont.)

Setting	Description
Max Version	The Max Version setting further defines the Windows Installer's search for installed versions to upgrade. The previously installed version's upgrade code must first match the value specified in the Upgrade Code setting. Once a matching upgrade code match is detected, the search can be narrowed by the minimum and/or maximum version specified and whether the versions are included or excluded (indicated in the Include Min Version and Include Max Version settings). This setting contains the maximum version number of the product that you want your current version to update. The setting can be populated in one of the following ways: ● If you added an upgrade entry to the Upgrade Paths tree by browsing for and selecting an .msi file or a `Setup.exe` file on your system, InstallShield uses the version of the .msi file in the Max Version setting. ● You can also type the maximum version in this setting in the format 0.00.0000 or 00.00.0000 (where the zeros represent the version number). ***Caution ▪*** *You must specify a value for at least one of the version settings: Min Version, Max Version, or both.*
Include Max Version	Specify whether you want to include the maximum version—specified in the Max Version setting—in the upgrade. If you want the Windows Installer to include the maximum version in the search for previously installed versions of your product on the target system, select Yes.

Table 9-10 ▪ Upgrade Path Settings (cont.)

Setting	Description
Language Identifiers	The Language Identifiers setting further defines the Windows Installer's search for installed versions to upgrade. The previously installed version's upgrade code must first match the value specified in the Upgrade Code setting. Once a matching upgrade code match is detected, the search can be narrowed by the languages specified and included or excluded (indicated in the Lang Search Criterion setting). This setting identifies the decimal language identifier or identifiers for the product version that you want to upgrade. ***To provide the decimal language identifier or identifiers for the product version that you want to upgrade:*** 1. Click the ellipsis button (**...**). The **Select Languages** dialog box opens. 2. Select the check box for each language version that you want to be updated by this upgrade. 3. Click **OK**. As an alternative, you can also type the decimal value or values for this setting. To indicate more than one language, separate the language identifiers with a comma (,). ***Note ▪*** *If you added an upgrade entry to the Upgrade Paths explorer by browsing for and selecting an .msi file on your system, InstallShield automatically uses the decimal language value for that .msi file by default for the Language Identifiers setting.*
Lang Search Criterion	Specify whether you want to include the language identifiers that are specified in the Language Identifiers setting in the Windows Installer's search for product versions previously installed on the target system. Leave this setting blank to include all languages.
Ignore Remove Failure	Specify whether you want the Windows Installer to continue installation of the current version if it fails to remove a previous version of your product that exists on the target system. If you select No, the Windows Installer aborts the installation of the current version if it fails to remove a previous version of your product that exists on the target system.

Table 9-10 ▪ Upgrade Path Settings (cont.)

Setting	Description
Migrate Feature States	Specify whether you want the upgrade to attempt to migrate the feature-selection states from an installed product version to the newer version. When installing the previous version of your product, an end user may have selected to install certain features, and not install others. During the upgrade to a new version of your product, these original selections can be retained as the default selections by selecting Yes for the Migrate Feature States setting. For example, if your original product contained **Feature1**, **Feature2**, and **Feature3**, and an end user installed only Feature1 and Feature3, by default **Feature1** and **Feature3** will be selected for installation when your upgraded setup is run. **Feature2** will, by default, remain unselected.

Update Notifications View

FlexNet Connect provides a mechanism that enables you to automatically notify your Web-connected end users when patches, updates, and product information for your application are ready for release. The Update Notifications view is where you enable automatic notification for your original installation.

Using FlexNet Connect is simple. When you enable FlexNet Connect, InstallShield includes the Software Manager in your installation. This desktop tool ships with your application and provides a tool for your users to view available updates. To publish updates to your users, you will use a Web-based management portal called the FlexNet Connect Publisher site.

FlexNet Connect includes a variety of options that can be purchased together for a complete solution, or separately for a customized solution. To learn more, visit the Revenera Web site.

Specify Application Data View

Application data includes any and all files you are adding to your installation. Files can be added in the Files view, through redistributables, or by scanning for dependencies.

Table 9-11 ▪ Views Under the Specify Application Data View

View	Description
Files	The Files view is the main way you add files to your installation project. This view behaves similarly to the Windows Explorer, allowing you to drag-and-drop files into your installation project.
Files and Features	Although you can select which feature you want to add your files to in the Files view, if you want to change which feature a file is associated with after the file has been added to your installation, you need to do so in the Files and Features view.

Table 9-11 ▪ Views Under the Specify Application Data View (cont.)

View	Description
Redistributables	Redistributables—InstallShield prerequisites, merge modules, and objects—enable you to add distinct pieces of functionality to your installation project. Examples of these types of functionality include Visual Basic run-time DLLs and the Microsoft C run-time libraries.
Dependencies	The Dependencies view includes three different scanners that add dependency files to your installation. These scanners include a Visual Basic project scanner, a static scanner that scans files already included in your project and adds their dependencies, and a dynamic scanner that scans a running application for any dependencies.

Files View

Files are the core of any installation. The main purpose of an installation program is to transfer files from the source medium to the target destination. The Files view is divided into four panes: The two left panes contain folders and the two right panes display the files located within those folders. Above these four panes is the Feature list. Before you add files to your installation, you need to select the feature with which you want to associate your files. The feature that appears in the box is the feature to which your files are added.

Table 9-12 ▪ Panes in the Files View

Pane	Description
Source computer's folders (Top Left)	The **Source computer's folders** pane is similar to the left pane in Windows Explorer. This pane contains folders located either locally or on a network. In this pane, you can navigate to the folder that contains the files that you want to add to your installation.
Source computer's files (Top Right)	The **Source computer's files** pane displays the files that are in the currently selected folder of the **Source computer's folders** pane. You can drag files from this pane to a location in one of the destination panes. The files that you drop into the bottom panes are added to your installation project.
Destination computer's folders (Bottom Left)	The **Destination computer's folders** pane contains a list of predefined destinations, as well folders that you have added to your project. To add a custom folder that you want the installation to create on target systems, right-click a folder in this view and then click New Folder. Some predefined folders are hidden by default. For information on displaying these folders, see Displaying Predefined Folders in the Files View. For information on adding files and folders to 64-bit locations on target systems, see Challenges of Supporting Both 32-Bit and 64-Bit Target Operating Systems.

Table 9-12 ▪ Panes in the Files View (cont.)

Pane	Description
Destination computer's files (Top Right)	The **Destination computer's files** pane displays of all the files that you have added to the currently selected destination folder. By right-clicking a file in this pane, you can cut, copy, paste, or delete the file, or you can edit its properties.

You can add files to your installation project in one of three different ways. Each of these methods is described below:

Table 9-13 ▪ Methods for Adding Files to a Project

Method	Description
Drag and Drop	The most straightforward way to add files to your installation project is by using the panes in the Files view. The top two panes in this view are functionally equivalent to Windows Explorer. The bottom two panes represent the destination for your files. Simply drag source files from the top pane to the destination folder in the bottom pane. ***Note ▪*** *If you drag and drop a folder from a source folder to the destination folder, InstallShield displays a dialog box that prompts you to indicate whether you want to make the folder a dynamic link to the folder.*
Dynamic File-Linking	Another way to add files to your installation is by linking to the contents of an entire folder, or to specific files in the folder. This method enables you to point to a specific folder, either locally or on a network, that contains files for your installation. Every time you build your installation, the selected contents of the folder are added to your feature. Additionally, you can use wild cards to filter which files are added to your installation.
Dependency Scanning	A third way to add files to your installation is accomplished through the Dependencies view. This view contains wizards that can scan your installation project or a running application for all dependency files and add them to your installation. These wizards can also be launched from the Tools menu.

Note ▪ *To refresh any of the panes in the Files view, click the pane you want to refresh and then press F5.*

Destination Folders

When you add files to your installation, you do so by placing them in a destination folder. The following predefined destination folders are provided by default in the Files view. Each one is dynamic, meaning that they do not rely upon hard-coded paths. Instead, the value for each destination folder is obtained from the operating system of the target machine.

Some predefined folders are hidden by default. For information on displaying hidden folders, see Displaying Predefined Folders in the Files View.

The following properties hold the fully qualified path to many of the folders on the end user's system.

Table 9-14 ▪ Predefined Destination Folders

Property	Description
AdminToolsFolder	Points to the folder where administrative tools are located.
AppDataFolder	This property holds the full path to the current user's Application Data folder. By default, this property is none.
CommonAppDataFolder	This is the full path to the folder containing application data for all users. A common path is: `C:\ProgramData`
CommonFiles64Folder	The value of this property is the full path to the 64-bit Common Files folder. For information on installing to 64-bit locations on target systems, see Challenges of Supporting Both 32-Bit and 64-Bit Target Operating Systems.
CommonFilesArmFolder	The value of this property is the full path to the Arm\Arm64 Common Files folder.
CommonFilesFolder	The value of this property is the full path to the Common Files folder.
DATABASEDIR	This property stores the destination for your installation's database files. You can set the initial value for DATABASEDIR in the General Information view and end users can modify this value at run time in the Database Folder dialog—if you display the Change button.
DesktopFolder	This property is used to hold the full path to the Desktop folder for the current user. If the setup is being run for All Users, and the ALLUSERS property is set, the DesktopFolder property should hold the full path to the All Users desktop folder.
FavoritesFolder	The FavoritesFolder property contains the full path to the Favorites folder for the current user.
FontsFolder	This property holds the full path to the Fonts folder.
GlobalAssemblyCache	This property contains the full path to the Global Assembly Cache (GAC).
INSTALLDIR	This property stores the destination folder for your installation. You can set an initial value for INSTALLDIR in the General Information view.

Table 9-14 ▪ Predefined Destination Folders (cont.)

Property	Description
IISROOTFOLDER	This property stores the root directory of the Web server on a target system. If you are using IIS functionality in your installation project and you have added a Web site, then IISROOTFOLDER is automatically added. ***Note ▪*** *All of the files that you add to the IISROOTFOLDER directory in the Files view are installed to the Web server's root directory on the target machine. If IIS is not on the target machine, the files are copied to the root folder.*
LocalAppDataFolder	The location of locally stored application data. A typical value of this property is: C:\Users*UserName*\AppData\Local
MyPicturesFolder	This property holds the full path to the current user's My Pictures folder.
PersonalFolder	This property holds the full path to the current user's Personal folder.
PrimaryVolumePath	This property is set to the path that is specified in the `PRIMARYFOLDER` property, which indicates the primary folder for the installation.
ProgramFiles64Folder	This property holds the full path to the 64-bit Program Files folder. For information on installing to 64-bit locations on target systems, see Challenges of Supporting Both 32-Bit and 64-Bit Target Operating Systems.
ProgramFilesArmFolder	This property holds the full path to the Arm\Arm64 Program Files folder.
ProgramFilesFolder	This property holds the full path to the 32-bit Program Files folder.
ProgramMenuFolder	This property is used to hold the full path to the Program menu for the current user. If the installation is being run for All Users, and the `ALLUSERS` property is set, the `ProgramMenuFolder` property should hold the full path to the All Users Programs menu.
SendToFolder	This property holds the full path to the current user's SendTo folder.
StartMenuFolder	This property is used to hold the full path the Start menu folder for the current user. If the installation is being run for All Users, and the `ALLUSERS` property is set, the `StartMenuFolder` property should hold the fully qualified path to the All Users program menu.
StartupFolder	This property is used to hold the full path to the Startup folder for the current user. If the setup is being run for All Users, and the `ALLUSERS` property is set, the StartupFolder property should hold the full path to the All Users program menu.
System16Folder	This property holds the full path to the folder containing the system's 16-bit DLLs.

Table 9-14 ▪ Predefined Destination Folders (cont.)

Property	Description
System64Folder	This property holds the full path to the 64-bit Windows system folder. For information on installing to 64-bit locations on target systems, see Challenges of Supporting Both 32-Bit and 64-Bit Target Operating Systems.
SystemArmFolder	This property holds the full path to the Arm\Arm64 Windows system folder.
SystemFolder	This property holds the full path to the 32-bit Windows system folder.
TempFolder	This property holds the full path to the Temp folder.
TemplateFolder	This property holds the full path to the current user's Templates folder.
WindowsFolder	This property holds the full path to the Windows folder.
WindowsVolume	This property holds the volume of the Windows folder. It is set to the drive where Windows is installed.

Files and Features View

The best time to associate files with a feature is when you first add those files to your setup project. After a file has been added, the only way to change the feature with which it is associated is through the Files and Features view.

Task ***To change the feature with which a file is associated:***

1. Click the feature that currently contains the file to display a list of all the files associated with that feature.
2. Select the file you want to move and drag it onto its new feature.

Note ▪ *You can also cut and paste files—both in large groups or individually—rather than using the drag-and-drop method.*

Redistributables View

The Redistributables view contains all of the InstallShield prerequisites, merge modules, and objects that are included with InstallShield.

InstallShield Prerequisites

An InstallShield prerequisite is an installation for a product or technology framework that is required by your product. You can add any of the existing InstallShield prerequisites to your installation projects.

Including InstallShield prerequisites in your project enables you to chain multiple installations together, bypassing the Windows Installer limitation that permits only one Execute sequence to be run at a time. The `Setup.exe` setup launcher serves as a bootstrap application that manages the chaining.

***Edition** ▪ InstallShield Premier Edition and InstallShield Professional Edition include the InstallShield Prerequisite Editor, a tool that enables you to define custom InstallShield prerequisites and to edit settings for any existing InstallShield prerequisites.*

InstallShield includes support for two types of InstallShield prerequisites:

- **Setup prerequisite**—The installation for this type of prerequisite runs before your installation runs.
- **Feature prerequisite**—This type of prerequisite is associated with one or more features. It is installed if the feature that contains the prerequisite is installed and if the prerequisite is not already installed on the system. Thus, if a feature has a condition that is not met on the target system, or if the end user chooses not to install the feature, the feature is not installed. As a result, none of its associated feature prerequisites are installed, unless the feature prerequisites are also associated with other features that are installed.

Merge Modules

A merge module (or .msm file) contains all of the logic and files needed to install distinct pieces of functionality. For example, many applications require Microsoft Visual Basic run-time .dll files. Instead of having to include the file in a feature and figure out its installation requirements, you can simply attach the Visual Basic Virtual Machine merge module to one of your project's features.

***Note** ▪ Many of the merge modules included in the Redistributables view are authored by Microsoft or another third party. InstallShield distributes these modules as a courtesy to assist you in creating your installation project. However, InstallShield cannot modify or fix any problems that may exist within third party-authored modules. You are encouraged to contact the vendor regarding issues with specific third party-authored modules.*

Objects

Like merge modules, objects contain logic and files needed to install distinct pieces of functionality. Some objects, such as the DirectX object included with InstallShield, require customization through a wizard. As soon as you add such an object to your installation, its customization wizard opens. You can either customize your object at the time you add it, or cancel the wizard and customize your object later by right-clicking the object and selecting Change Objects Settings.

Redistributables Gallery

Because the file size of many of the redistributables is so large, some that are available for use in your projects are not added to your computer when you install InstallShield. However, these redistributables are still available for download from the Internet to your computer.

Configurable Merge Modules

A configurable redistributable is a merge module or an object that has at least one row in the `ModuleConfiguration` table that is referenced by at least one row in the `ModuleSubstitution` table. This enables you to change a value in the redistributable. When you select a configurable module in the Redistributables view, the Merge Module Configurable Values dialog box is displayed to enable you to configure the module at the time you add it. To customize the merge module later, right-click it and select **Configure merge module**.

Working with the Redistributables View

The Redistributables view consists of the following elements:

- A row of buttons and other controls
- A group box area (below the row of buttons)
- A list of redistributables
- A details pane, which shows information about the selected redistributables

The following table describes all of the buttons and other controls that are displayed in the Redistributables view.

Table 9-15 ▪ Controls in the Redistributables View

Name of Control	Icon	Description
Expand All Groups		Shows all of the rows in the groups if you are using groups to organize the rows in a hierarchical format.
Collapse All Groups		Hides all of the rows in the groups if you are using groups to organize the rows in a hierarchical format.
Show Details		Shows or hides the right pane in this view, which shows details about the InstallShield prerequisite, merge module, or object that is selected in the list of available redistributables. This details pane provides information such as which files a redistributable installs.
Show Group Box		Shows or hides the group box area below the row of buttons in this view.
Refresh		Refreshes the list of redistributables.
Type a string to filter by		Dynamically filters the redistributables that are displayed in the Redistributables view according to the string that you specify in this search box. As you type a string in this box, InstallShield hides all of the rows that do not contain it.
Redistributables View Help		Displays the help for the Redistributables view.

Table 9-15 ▪ Controls in the Redistributables View (cont.)

Name of Control	Icon	Description
Drag a column header here to group that column		Use this group box area to group rows in the view. The view supports multiple levels of grouping simply by dragging the column headings and dropping them onto the group box. InstallShield displays the rows in the view hierarchically according to column arrangement in the group box. To learn more, see Working with the Group Box Area in Various Views.

The following table describes each of the columns in the Redistributables view.

Table 9-16 ▪ Columns in the Redistributables View

Column	Description
Check Box	This column shows a check box for each redistributable. To add a redistributable to your project, select its check box.
Type/Status	This column shows an icon that represents the type of redistributable, as well as its status. For details about each icon, see Managing the Redistributables Gallery.
Name	This column shows the name of each redistributable.
Version	This column shows the version number of the redistributable.
Type	This column lists the type of redistributable.
Location	This column indicates whether the redistributable is installed locally on your machine or it needs to be downloaded. For more information, see Downloading Redistributables to Your Computer.

Dependencies View

A file often relies on functions in other files to perform a task. However, you may not be aware of all these other files—known as *dependencies*—when you include your application's files in your installation project. InstallShield offers the following scanning wizards to assist you in identifying and working with these files. You can access these scanners in the Dependencies view:

Table 9-17 ▪ Dependency Scanners in InstallShield

Scanning Option	Function
Perform Static Scanning	The Static Scanning Wizard looks at portable executable files (for example, .exe, .ocx, .com, .tlb, .hlp, and .chm) in your project and checks for dependencies that they may require.

Table 9-17 ▪ Dependency Scanners in InstallShield (cont.)

Scanning Option	Function
Perform Dynamic Scanning	The Dynamic Scanning Wizard monitors your system while an executable file is running and checks for .dll or .ocx files that may be required by the executable file.

If you use the Static and Dynamic scanning wizards, you can specify files that you want to be included or excluded automatically any time that you perform a static or dynamic scan through InstallShield. For more information, see Filtering Files in Dependency Scanners.

Configure the Target System View

Every installation changes the target system in some way. The simplest installations merely copy files. More in-depth installations make registry changes, edit .ini files, create shortcuts, or make other changes to configure the target system. InstallShield provides the following subviews for making advanced configurations:

Table 9-18 ▪ Views Under the Configure the Target System View

View	Description
Shortcuts/Folders	Shortcuts offer quick access to your installed application. You can create shortcuts and folders on the desktop, the Start menu, and various other locations.
Registry	Through the Registry view, you can create registry entries or import existing registry data into your installation.
ODBC Resources	In the ODBC Resources view, you can add drivers, translators, and data sources to one or more of your application's features. Select from installed ODBC resources or add new ones to the list, associate them with features, and then customize their attributes.
INI File Changes	This view enables you to edit an .ini file on the target system.
File Extensions	If your product requires file extension associations, you can create these associations in the File Extensions view.
Environment Variables	If you need to create, modify, or remove environment variables from the target system, you can define their properties in the Environment Variables view.
Internet Information Services	The Internet Information Services view enables you to create and manage an IIS Web site, applications, and virtual directories on the target system.
Component Services	The Component Services view enables you to manage COM+ server applications and components for your installation package.
Services	You can use the Services view to specify information about a service that you want to install during installation and remove during uninstallation.

Shortcuts/Folders View

The Shortcuts/Folders view provides a simple, visual way to create shortcuts to your application on the target system. This view contains a Shortcuts explorer that shows a set of predefined destination folders under which you can create shortcuts and subfolders. InstallShield offers the following standard shortcut destinations.

Table 9-19 ▪ Standard Shortcut Destinations

Shortcut Destination	Description
Programs Menu and Startup	The Programs Menu and Startup folders are located on the Start menu. The Programs Menu folder is the industry-standard place for installing product shortcuts. The Startup folder should contain shortcuts to only those items that need to be launched whenever Windows starts.
Send To	The Send To folder is available when an end user right-clicks a file; the context menu that is displayed contains a command called Send To. If you create a shortcut for your program in this folder, an end user can click any file and send it to your product. For example, you might want your end users to be able to open an HTML page in Notepad. If you created a shortcut to Notepad in the Send To folder in the Shortcuts/Folders view, end users could right-click an HTML file and click Notepad from your Send To menu. The source file for that page opens in Notepad. ***Note ▪*** *A shortcut created in the Send To Folder is set to the SendTo folder for the user running the installation. This is because Windows Installer sets the SendToFolder property to the full path to the SendTo folder for the current user.* *Shortcuts in the Send To folder cannot be advertised.*
Desktop	The Desktop folder contains shortcuts for the end user's desktop. When you create a shortcut in the Desktop folder, your product's icon is displayed on the end user's desktop. ***Note ▪*** *The desktop is the most visible place to put a shortcut, but too many shortcut icons can clutter the end user's desktop.*

When you add a new shortcut, InstallShield adds it as a new node to the Shortcuts explorer in this view. The new shortcut node is displayed with the same icon that is selected in the Icon File setting for the shortcut—which is the icon that is used when the shortcut is added to target systems at run time—unless the shortcut is for a preexisting file on the target system.

If the shortcut is for a preexisting file, the icon file is not known until run time. Therefore, InstallShield shows the following icon for each shortcut in the Shortcuts explorer, instead of displaying the icon that is used at run time on target systems.

InstallShield also uses this icon for a shortcut in the Shortcuts explorer if the icon file does not contain an icon.

Note ▪ *You cannot create shortcuts to a dynamically linked file. For more information, see Limitations of Dynamic File Linking.*

For details about each of the settings that you can configure for shortcuts and folders, see:

- Shortcut Settings
- Folder Settings

Shortcut Settings

The settings for a shortcut are organized into the following main categories in the Shortcuts/Folders view:

- Appearance
- Behavior
- General

Appearance

Use a shortcut's Appearance settings to specify details such as the description and the icon for the shortcut.

Table 9-20 ▪ Appearance Settings

Setting	Description
Display Resource	If you are preparing an installation for a multilingual application and you are separating language-neutral portable executable files from .mui files for the display name of your shortcut, use the following settings: • **Display Resource**—Click the ellipsis button (...) in this setting if you want to browse to the DLL file that contains the multilingual user interface (MUI) manifest. InstallShield lists the path and file name in this setting if you do either of the following: you select a DLL file by browsing for it, or you manually enter a path and file name in the Display Resource DLL setting. InstallShield also lists the resource index that is specified in the Display Resource Index setting. • **Display Resource DLL**—If you want to manually specify the path and file name of the DLL file that contains the MUI manifest, enter it. You can include Windows Installer directory properties—for example, **[INSTALLDIR]MyResource.dll**—instead of hard-coded directory paths. If you click the ellipsis button in the Display Resource setting to browse to the DLL file, InstallShield uses the format [#filekey] in the Display Resource DLL setting to identify the DLL file. • **Display Resource Index**—Specify the display name index for the shortcut. This must be a non-negative number. ***Note ▪*** *If you specify a DLL, you must also enter a value for the Display Resource Index setting.* These settings enable you to separate language-neutral portable executable files from .mui files, which contain all of the language-dependent resources, and later add resources for additional languages without having to recompile or relink the application. Windows Vista and later and Windows Server 2008 and later include support for the display resource. Earlier systems ignore it.
Description	Enter a description of the shortcut. The text that you enter is displayed as a tooltip when the end user places the mouse pointer over the shortcut. It is also displayed in the Description field of the shortcut's Properties dialog box.

Table 9-20 ▪ Appearance Settings (cont.)

Setting	Description
Description Resource	If you are preparing an installation for a multilingual application and you are separating language-neutral portable executable files from .mui files for the description of your shortcut, use the following settings: • **Description Resource**—Click the ellipsis button (...) in this setting if you want to browse to the DLL file that contains the multilingual user interface (MUI) manifest. InstallShield lists the path and file name in this setting if you do either of the following: you select a DLL file by browsing for it, or you manually enter a path and file name in the Description Resource DLL setting. InstallShield also lists the resource index that is specified in the Description Resource Index setting. If this setting contains a value, the value for the Description setting is ignored. If you leave this setting blank, Windows Installer uses the value for the Description setting. • **Description Resource DLL**—If you want to manually specify the path and file name of the DLL file that contains the MUI manifest, enter it. You can include Windows Installer directory properties—for example, **[INSTALLDIR]MyResource.dll**—instead of hard-coded directory paths. If you click the ellipsis button in the Description Resource setting to browse to the DLL file, InstallShield uses the format [#filekey] in the Description Resource DLL setting to identify the DLL file. • **Description Resource Index**—Specify the description index for the shortcut. This must be a non-negative number. ***Note ▪*** *If you specify a DLL, you must also enter a value for the Description Resource Index setting.* These settings enable you to separate language-neutral portable executable files from .mui files, which contain all of the language-dependent resources, and later add resources for additional languages without having to recompile or relink the application. Windows Vista and later and Windows Server 2008 and later include support for the description resource. Earlier systems ignore it.

Table 9-20 ▪ Appearance Settings (cont.)

Setting	Description
Icon	To specify the icon that you want to be displayed for the shortcut, use the following settings: • **Icon**—Specify the file that contains the icon for the shortcut that you are creating. You must specify an .ico file or the executable file (.dll or .exe) that contains the icon resource. InstallShield lists the icon path and index in the Icon setting if you do either of the following: you select a file by clicking the ellipsis button (...) in this setting and browsing for it, or you manually enter a path and file name in the Icon File setting. • **Icon File**—If you want to manually specify the path and name of the file that contains the icon, enter it. • **Icon Index**—If the icon file that you specify contains more than one icon resource, enter the index in this setting. A nonnegative integer refers to the order of the icon resources in the executable file. For example, 0 refers to the first icon in the file, 1 refers to the second icon, and 2 refers to the third icon. Use a negative number to refer to a specific resource ID. For example, the icon index -12 points to the icon with a resource ID of 12.

Behavior

Use a shortcut's Behavior settings to specify details such as the target and keyboard shortcut.

Table 9-21 ▪ Behavior Settings

Setting	Description
Target	Enter the path to the file on the target system that should be launched when end users launch the shortcut. Use Windows Installer directory properties—for example, **[INSTALLDIR]File.exe**—instead of hard-coded directory paths. As an alternative to manually entering a value, you can click the ellipsis button (...) to browse to the shortcut target.

Table 9-21 ▪ Behavior Settings (cont.)

Setting	Description
Arguments	Enter the command-line arguments for the shortcut. The arguments are added to the Target value on the shortcut's Properties dialog box on the target system. These arguments work in the same way as any other command-line arguments. For example, you can link a file to an executable file or cause an executable file to run silently by passing command-line arguments. ***Note ▪*** *Verify that the syntax is correct because InstallShield does not do this.* ***Tip ▪*** *Use* **%1** *in the argument for the selected file name. For example, if the end user right-clicks the file* `C:\File.ext` *and* `-p %1` *is the argument for this shortcut, the command-line argument becomes* `-p C:\File.ext`*. In some cases, it is necessary to enclose the %1 argument in quotation marks—as in* **"%1"***—to correctly handle file names that contain spaces.*
Working Directory	Enter the working directory for the shortcut target, or click the ellipsis button (...) to select or create a directory. The directory that you specify is displayed as the **Start in** field on the shortcut's Properties dialog box on the target system. The working directory is the default directory that is displayed in standard file-opening and file-saving dialog boxes, as well as the current directory used by the product. ***Project ▪*** *For example, if you want your working directory to be set to a subdirectory of* `INSTALLDIR` *called* **Files***, select* ***[INSTALLDIR]*** *from the list and add the subdirectory name* **`Files`** *to the end of it. When you are finished, it should read* ***[INSTALLDIR]Files****.*
Hot Key	This setting contains the decimal value of the keyboard shortcut that is assigned to your shortcut. You can assign a keyboard shortcut to your product's shortcut so that end users can press the appropriate hot keys to launch the shortcut. If you want InstallShield to calculate the decimal value of the keyboard shortcut, click the ellipsis button (...) in this setting. For more information, see Specifying a Keyboard Shortcut for Accessing a Shortcut. ***Caution ▪*** *It is recommended that you avoid configuring keyboard shortcuts for your shortcuts because they may conflict with existing keyboard shortcuts on the target system.*

Table 9-21 ▪ Behavior Settings (cont.)

Setting	Description
Run	Select the style of window that the target file should use when end users launch the shortcut. Available options are: • **Normal Window**—The file is launched in a standard-sized window. • **Maximized Window**—The file is launched in full-screen view. • **Minimized Window**—The file is launched in a minimized window, visible only on the taskbar.
Run As Administrator	Select to set the shortcut to "Run As Administrator". Available options are: • **Yes**—Enables the shortcut to Run As Administrator. • **No**—To not to set the shortcut to Run As Administrator. ***Note ▪*** *An installer should run in an Elevated mode in order to set the shortcut with Run as Administrator.* *Run as Administrator will be disabled for Advertised Shortcut and Internet Shortcut.*

General

Use a shortcut's General settings to specify details such as the feature that contains the shortcut.

Table 9-22 ▪ General Settings

Setting	Description
Feature	This setting indicates the feature or features with which the shortcut is associated.
Comments	Enter comments for this shortcut. Comments are saved in the project file for your reference and are not displayed to the end user.

Folder Settings

The following setting is available for a new folder that you create in the Shortcuts/Folders view.

Table 9-23 ▪ Folder Settings

Setting	Description
Description	Enter a description of the folder. InstallShield adds this folder to the Always Install feature. Therefore, the folder is always installed, even if the shortcut within the folder is not installed.

Tile Configuration Settings

The settings for tile configurations are organized into the following main categories:

- General
- Appearance

General

Use the tile configuration's General setting to view the path of the .exe file that is associated to the tile.

Table 9-24 ▪ General Settings

Setting	Description
Target	Displays the path of the .exe file (for example, **[INSTALLDIR]File.exe**) that is associated to the tile.

Appearance

Use a tile configuration's Appearance settings to specify tile customizations such as the background color of the tile.

Table 9-25 ▪ Appearance Settings

Setting	Description
Text Color	Choose a foreground text color that is used to display the app name on a medium-sized tile. Available drop-down options are: ● **Light**—Sets the text color that is displayed on a medium-sized tile to be light (white text). ● **Dark**—Sets the text color that is displayed on the medium-sized tile to be dark (black text). ***Note ▪*** *Only a medium-sized tile (150x150) has room to display a name on the tile. Also, in order for the* ***Text Color*** *setting to have an effect, the* ***Show Name on Medium Tile*** *setting must be* ***Yes****.*
Background Color	Select a color to be used as the background color of the tile. A drop-down selection provides several predefined colors, or an ellipsis button (...) lets you choose additional colors or specify an RGB hex string color.
Small Tile Image	Specify a relative path to a small-sized (70x70) tile background image. ***Important ▪*** *If you choose to use custom images, you must specify an image for both the small tile image (70x70) and the medium tile image (150x150) and they must be added to the project manually. If you specify only one, the default Windows styling for the app icon and background is applied to the tile. Also, if you provide scaled, high-contrast, or localized images, you need to manually create a* `resources.pri` *file and place it in the same directory as the app's .exe file and its* `.visualelementsmanifest.xml` *file. For more information about using images or creating a* `resources.pri` *file, refer to the following MSDN article: Desktop App Tiles on the Start Screen.*
Medium Tile Image	Specify a relative path to a medium-sized (150x150) tile background image. ***Important ▪*** *If you choose to use custom images, you must specify an image for both the small tile image (70x70) and the medium tile image (150x150) and they must be added to the project manually. If you specify only one, the default Windows styling for the app icon and background is applied to the tile. Also, if you provide scaled, high-contrast, or localized images, you need to manually create a* `resources.pri` *file and place it in the same directory as the app's .exe file and its* `.visualelementsmanifest.xml` *file. For more information about using images or creating a* `resources.pri` *file, refer to the following MSDN article: Desktop App Tiles on the Start Screen.*

Table 9-25 ▪ Appearance Settings (cont.)

Setting	Description
Show Name on Medium Tile	Specify whether the app name should be shown on a medium-sized tile (150x150). Available options are: • **Yes**—Display the app name on the tile when a medium-sized tile is used. • **No**—Do not display the app name on the tile. ***Note ▪*** *This setting has no effect when a small-sized tile is used.*

Registry View

InstallShield makes the task of modifying the end user's registry more familiar with the InstallShield Registry view. Use this view to create keys and values in a way that is similar to how you use the Windows Registry Editor.

Additionally, the installer automatically creates certain registry entries based on values you provide for your General Information settings. These "informational keys" are required by Windows logo guidelines.

To see all of the registry data in your installation project, select All Application Data in the Feature list at the top of the view. You can modify or delete registry data, but cannot add new keys or values when this option is selected.

Tip ▪ *If you are creating a 64-bit installation and you want to install registry entries to 64-bit registry locations (under HKEY_LOCAL_MACHINE\Software instead of HKEY_LOCAL_MACHINE\Software\Wow6432Node), add the entry to the SOFTWARE (64-Bit) node, or a subnode. Note that a 64-bit installation cannot be run on 32-bit target systems. To learn more, see Challenges of Supporting Both 32-Bit and 64-Bit Target Operating Systems.*

To see how a 32-bit application views the registry on a 64-bit system, launch the 32-bit version of the Registry Editor (the `regedit.exe` *file in the SysWOW64 folder).*

ODBC Resources View

One of the more complex areas of system configuration involves setting up ODBC drivers, data source names (DSNs), and translators. The ODBC resource must be properly registered on the system with all of the required attributes and, in the case of drivers and translators, install the necessary files, including any setup DLL. This process is greatly simplified in the ODBC Resources view, in which you can simply check off the drivers, data sources, and translators installed on your development system.

All of the ODBC drivers, data source names (DSNs), and translators registered on your system are displayed in the ODBC Resource view. DSNs are shown as children of their associated driver. Expand the tree to view all of the existing ODBC resources. Like most of the data in your setup project, ODBC resources must be associated with a feature. Then, when that feature is installed, the ODBC resource is installed as a part of the overall feature.

Tip ▪ *The ODBC Resources view is exclusively for installing ODBC-related resources. To install the core ODBC files, select the MDAC merge module in the Redistributables view.*

ODBC Resource Settings

There is no universal list of ODBC resource attributes and their possible values. Check with the file's vendor or author if you are unsure about what to specify for its installation. Nonetheless, a few of the attributes are common to each type of ODBC resource, as described below.

***Note** ▪ You can add your own ODBC attribute by clicking in the last row of the grid in the ODBC Resources view and specifying the property and value.*

Drivers

In addition to the required attributes below, the driver must have a name in the tree. This name is registered as the driver's description. The driver's name cannot be localized, which means that you cannot translate it through the Text and Messages view and the same value must be used regardless of the system's language.

Table 9-26 ▪ ODBC Driver Attributes

Attribute	Expected Value
Driver DLL	Enter the path to the DLL file that is on the development system and that serves as the ODBC driver, or click the ellipsis button (...) to browse to the file.
Setup DLL	Enter the path to the setup DLL on the development system, or click the ellipsis button (...) to browse to the file. If the driver file is also the setup DLL, you can leave this setting blank.

Data Sources

In addition to the required attributes below, the DSN must have a name in the tree. This name is registered as the DSN's description. The DSN's name cannot be localized, which means that you cannot translate it through the Text and Messages view and the same value must be used regardless of the system's language.

Table 9-27 ▪ ODBC Data Source Attributes

Attribute	Expected Value
Registration	Specify whether you want to use a system data source or a user data source. • **System data source**—The data source is available to all users on the system. • **User data source**—The data source is registered only for the current user.

Translators

In addition to the required attribute below, the translator must have a name in the tree. This name is registered as the translator's description. The translator's name cannot be localized, which means that you cannot translate it through the Text and Messages view and the same value must be used regardless of the system's language.

Table 9-28 ▪ ODBC Translator Attributes

Attribute	Expected Value
Translator DLL	Enter the path to the DLL file that is on the development system and that serves as the translator, or click the ellipsis button (...) to browse to the file.
Setup DLL	Enter the path to the setup DLL on the development system, or click the ellipsis button (...) to browse to the file.

Note ▪ *You cannot add an attribute to a translator.*

INI File Changes View

Editing .ini files found on the target system can be hazardous, especially if those .ini files are used by the operating system for standard functionality, such as `Boot.ini`. With InstallShield, .ini file changes can be performed right in the InstallShield interface. Editing an .ini file involves three tasks, which are described below.

Table 9-29 ▪ Tasks for Editing an .ini File

Task	Description
Adding an .ini File	The first step in creating an .ini file change is to create a reference to the file you want to edit. In order to do this you need to know the name and location of the file you want to edit. If the file is not in the location you specify, no changes to this file can be made. You can also import an .ini file.
Specifying a Section in an .ini File	Once you have specified the .ini file you want to edit, the next step is to specify which section of that file you want to change. INI files are divided into sections, each section containing keywords. Sections are identified by the square brackets surrounding them—[SectionName], for example.
Specifying a Keyword and its Value in an .ini File	INI File keywords are the lowest level of organization in an INI file. These keywords store data that must persist between uses of an application.

For details about each of the settings that you can configure for .ini file changes, see:

- Settings for .ini Files
- Settings for .ini File Keywords

Settings for .ini Files

When you add an .ini file change to your installation, target and feature settings should be configured. To access these settings, open the INI File Changes view, and in the INI Files explorer, select the .ini file.

Table 9-30 ▪ .ini File Settings

Setting	Description
Feature	Select the feature with which you want this .ini file change to be associated. When the selected feature is installed, the change to the .ini file occurs. If the selected feature is not installed, the change does not take place.
Target	Specify the location of the .ini file on the target system. Instead of entering a hard-coded path, you can select a Windows Installer folder property from the drop-down list. ***Tip ▪*** *Do not separate subfolders from the folder property with a backslash, but separate further levels of subfolders with a backslash—for example,* `[ProgramFilesFolder]MyCompany\Subdirectory`*.*

Settings for .ini File Keywords

A keyword is the lowest level of organization in an .ini file. The keywords store data that must persist between uses of an application. The settings for keywords are described below.

Table 9-31 ▪ .ini File Keyword Settings

Setting	Description
Action	Select the action that you would like to perform. Valid options are: • **Replace Old Value**—The existing value is replaced with the value that you enter in the Data Value setting. If the value does not exist, the installation creates it. • **Do Not Overwrite**—Your value is added to the .ini file if the keyword does not already exist. No changes are made if the keyword is already present in the .ini file. • **Append Tag**—The installation appends your new value to that of the existing value. The new value is separated from the existing value by a comma. If the keyword to which you would like to append a tag does not exist, no changes are made. • **Remove Whole Value**—The keyword and its value are both removed from the .ini file. If the specified keyword does not exist, no changes are made. If you select this option, leave the Data Value setting blank. • **Remove Tag**—Multiple values for a keyword are known as *tags*. Tags are separated by commas. To remove a tag from a keyword's value, select this option. In the Data Value setting, enter the tag that you want to be removed.
Data Value	Enter the value for the keyword. If you are adding or appending a value, enter the new value. If you are removing a tag, enter the tag that you would like to remove. If you are removing the keyword and value, leave this setting blank. You can use Windows Installer properties in your keyword's value. To do this, surround the property with square brackets—for example, `[INSTALLDIR]`.

File Extensions View

File extensions enable you to link a certain type of file to your product. When that file is double-clicked, your product is launched, and your product opens that file. When you open a text (.txt) file, you are actually sending a message to the operating system, telling it to launch Notepad so that you can view the contents of that text file. If you want to provide similar functionality for your product and its files, you can create a file extension association.

For details about each of the settings that you can configure for a file association, see File Extension Settings.

File Extension Settings

File extension settings enable you to specify detailed information about the associated file type. To access these settings, open the File Extensions view, and in the File Extensions explorer, select the file extension that you want to edit.

Each file extension setting is described below:

Table 9-32 ▪ File Extension Settings

Setting	Description
Feature	Select the feature that you want this file type association to be a part of. If the selected feature is installed, the file type association will be made.
File	Select from the list the executable file for which you are creating a file extension association. If you have not yet added the necessary file to your installation project, it is not displayed in the list. ***Important ▪*** *It is not possible to create a file extension association for a dynamically linked file. If you want to configure an association, the file must be included in your project statically.*
Command	Enter the text that you want displayed on the context menu for the Open verb. If you leave this setting blank, the word `Open` appears as the first command on the context menu for this type of file. If you prefer it to contain the text `Edit with MyProduct`, enter **`&Edit with MyProduct`** in this setting.
Arguments	Enter any command-line arguments that you want to pass to your product when a file of this type is opened. You can use **`%1`** in your argument to pass the currently selected file name as part of the argument. For example, **-p %1** might be resolved to **-p C:\MyFile.ext**.
ProgID	Enter the progID of the product with which you are associating the file extension.
ProgID Description	This description is registered as the default value for the ProgID in the target system's registry.

Table 9-32 ▪ File Extension Settings (cont.)

Setting	Description
Icon	To specify the icon that you want to use for files of this type, use the following settings: • **Icon**—Click the ellipsis button (...) in this setting if you want to browse to the .ico file or the .exe file that contains the icon resource. InstallShield lists the icon path and index in this setting if you do either of the following: you select a file by browsing for it, or you manually enter a path and file name in the Icon File setting. • **Icon File**—If you want to manually specify the path and name of the file that contains the icon, enter it. • **Icon Index**—If the icon file that you specify contains more than one icon resource, enter the index in this setting. A nonnegative integer refers to the order of the icon resources in the executable file. For example, 0 refers to the first icon in the file, 1 refers to the second icon, and 2 refers to the third icon. Use a negative number to refer to a specific resource ID. For example, the icon index -12 points to the icon with a resource ID of 12.

Environment Variables View

Environment variables are name and value pairs that can be set on the target system with your installation program and can be accessed by your application and by other running programs.

Using InstallShield, you can create, set (or modify), and remove environment variables on the target system via your installation program. To specify environment variable properties, go to the Environment Variables view in the View List.

Environment Variable Settings

By specifying environment variable settings, you indicate how you want to affect existing variables on the target system or create new variables. Each environment variable setting is described below:

Table 9-33 ▪ Environment Variable Settings

Setting	Description
Feature	Select the feature with which you want to associate this environment variable. The environment variable creation, modification, or removal takes place on the target system when the feature that is associated with the environment variable is installed. ***Note ▪*** *To ensure that this environment variable is created, modified, or removed when your application is installed, select Always Install. If you want the environment variable to be associated with only a particular feature, select that feature instead.*
Value	Enter the path or value for this environment variable. You can use a predefined folder as part of this value—such as `[INSTALLDIR]bin`. To enter multiple paths, separate the paths with a semicolon (;). ***Note ▪*** *If you select Remove for the On Install setting, InstallShield clears any value entered in the Value setting, and the Value setting becomes read-only.*
On Install	Indicate the action that you want to take place when the associated feature is installed to the target system. Available options are: • **Create**—The installation creates the specified environment variable on the target system—if it does not already exist—and sets its value. • **Set**—In conjunction with the Placement setting, this option sets the value of an existing environment variable. If you select this option and the environment variable does not already exist on the target system, the installation creates it and then sets its value. If the environment variable exists on the target system, the installation sets its value. • **Remove**—The installation removes the specified environment variable from the target system. ***Note ▪*** *If you select Remove for the On Install setting, any value entered in the Value setting is cleared, and the Value setting becomes read-only.*

Table 9-33 ▪ Environment Variable Settings (cont.)

Setting	Description
Placement	Indicate the placement of the value in the Value field relative to the specified environment variable's existing value. Available options are: • **Append**—The installation appends the new value that is entered in the Value setting to the end of the existing value. • **Prefix**—The installation adds the new value that is entered in the Value setting to the beginning of the existing value. • **Replace**—The installation replaces the value of the specified environment variable with the new value that is entered in the Value setting. ***Note ▪*** *If you select Create for the On Install setting and the specified environment variable already exists on the target system, the Placement setting indicates how the new value should be added to the existing environment variable's value or if it should replace the existing environment variable's value. However, if—in this scenario—the specified environment variable does not exist on the target system, it is created and the Placement setting is ignored.*
On Uninstall	Indicate whether the environment variable should be updated when the associated feature is uninstalled. Available options are: • **Remove**—If you select Append or Prefix for the Placement setting and the environment variable value on the target system contains the specified value at uninstallation time, only that value is removed from the existing variable's value. If you select Replace for the Placement setting, and if either of the following conditions are true at uninstallation time, the entire environment variable is removed: the value on the target system matches the specified value, or the value on the target system is empty. In all other cases, the environment variable and its value are left as is on the target system. • **Leave**—The environment variable and, if it is available, its value are left as is on the target system.
Type	Indicate whether the environment variable name refers to a system or user environment variable. Available options are: • **System**—The installation creates, modifies, or removes the specified system environment variable. • **User**—The installation creates, modifies, or removes the environment variable from the end user's environment. The specified environment variable is available only to the end user who is logged on at the time of installation.

Internet Information Services View

The Internet Information Services view enables you to create and manage a new IIS Web site, applications, and virtual directories.

***Edition** ▪ The Express edition of InstallShield includes support for installing only one Web site per installation.*

The following editions of InstallShield support the creation of more than one Web site per installation. These editions also include support for managing IIS application pools and Web service extensions:

- *Premier*
- *Professional*

In addition, the Premier edition of InstallShield includes an IIS scanner that checks an existing IIS Web site and records data about the settings for the Web site, its virtual directories, its applications, and its application pools. You can import that IIS data into the Internet Information Services view in InstallShield Premier Edition. Once you have imported the IIS data into a project, you can use the Internet Information Services view to make changes to the IIS settings as needed.

The Internet Information Services view settings are organized into the following main areas:

- Web Site Settings
- Application and Virtual Directory Settings

The following table describes the buttons that are displayed above the settings in the Internet Information Services view. The buttons are displayed if a Web site, application, or virtual directory is selected in the center pane.

Table 9-34 ▪ Controls in the Internet Information Services View

Name of Control	Icon	Description
Categorized		Sorts the settings according to categories.
Alphabetical		Sorts the settings alphabetically.

Web Site Settings

***Edition** ▪ The Express edition of InstallShield includes support for installing only one Web site per installation.*

The following editions of InstallShield support the creation of more than one Web site per installation. These editions also include support for managing IIS application pools and Web service extensions:

- *Premier*
- *Professional*

In addition, the Premier edition of InstallShield includes an IIS scanner that checks an existing IIS Web site and records data about the settings for the Web site, its virtual directories, its applications, and its application pools.

You can import that IIS data into the Internet Information Services view in InstallShield Premier Edition. Once you have imported the IIS data into a project, you can use the Internet Information Services view to make changes to the IIS settings as needed.

Use the Web Sites item in the Internet Information Services view to add and delete a Web site, and to configure system-wide settings for the Web server.

Web Server Settings

When you select the Web Sites explorer in the Internet Information Services view, the following Web server settings are displayed.

Table 9-35 ▪ Web Server Settings

Setting	Description
Restart Web Server After Configuring IIS (IIS 6 and earlier only)	To restart IIS after each time the installation is done making IIS changes to the system, select Yes. Some applications may require IIS to be restarted. This setting applies to IIS 6 and earlier. IIS 7 ignores this setting.

Table 9-35 ▪ Web Server Settings (cont.)

Setting	Description
SSIEnableCmdDirective Registry Value	Specify how you want the SSIEnableCmdDirective registry value for the HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W3SVC\Parameters registry key to be set on the target system. The SSIEnableCmdDirective registry value controls whether the Web server allows the #exec CMD directive of server-side includes (SSIs) to be used to execute shell commands. Valid options are: • **Ignore**—Do not change the SSIEnableCmdDirective registry value on the target system. This is the default option. • **FALSE (0)**—Set the SSIEnableCmdDirective registry value on the target system to 0. This prevents the #exec CMD directive of server-side includes to be used to execute shell commands. Note that if you select this value and an IIS Web server has applications that rely on #exec CMD directives, those applications may stop working properly after your installation project's Web site and virtual directory are installed. • **TRUE (1)**—Set the SSIEnableCmdDirective registry value on the target system to 1. This allows the #exec CMD directive of server-side includes to be used to execute shell commands. If you select the FALSE or TRUE options, InstallShield stores the value—either 0 for FALSE or 1 for TRUE—in the `INSTALLSHIELD_SSI_PROP` property. Because of security concerns, the default SSIEnableCmdDirective registry value is FALSE (0); the FALSE (0) value prevents end users from running unauthorized server-side executable files. ***Note ▪*** *If your product is uninstalled from a target system, the SSIEnableCmdDirective registry value is not changed, even if its value was changed during installation.* For more information, see Specifying Whether a Web Server Should Allow the CMD Command to Be Used for SSI #exec Directives.

Note ▪ *The aforementioned Web server settings are not updated on a target system at installation run time if no IIS items (Web site, applications, or virtual directories) are installed.*

Web Site Settings

When you select a Web site in the explorer, many settings are displayed. The Web site settings are organized into several main categories:

- Identification
- General
- Home Directory

- Application Settings
- Security
- Advanced

Identification Settings

The following settings are available in the Identification area for a Web site in the Internet Information Services view.

Table 9-36 ▪ Identification Settings for a Web Site

Setting	Description
Name	Enter a name for the Web site.
IP Address	To target a specific IP address, enter it. As an alternative, you can leave an asterisk (*), which is the default value. An asterisk or a blank value for this setting indicates that any unused IP address should be used.
TCP Port	The TCP Port setting for an IIS Web site indicates the port number on which the service is running on the target machine. Some versions of the IIS Web server can host multiple Web sites. Each Web site is associated with a unique port number. If you do not know what the port number on the target system should be, you can enter 0 for this setting. To learn which port and site numbers are used for the Web site when it is installed, see Configuring the TCP Port and Site Numbers.
Host Header Name	To specify the host header name that identifies the IIS Web site that is installed during your installation, type it in this box. For example: `www.mycompany.com` Host headers (also known as domain names) enable you to assign more than one Web site to an IP address on a Web server.
Site Number	The Site Number setting indicates the number in the path at which the Web site will be created (that is, w3svc/3). The default value is 0. If you do not know what the site number on the target system should be, you can enter 0 for this setting. To learn which port and site numbers are used for the Web site when it is installed, see Configuring the TCP Port and Site Numbers.

General Settings

The following settings are available in the General area for a Web site in the Internet Information Services view.

Table 9-37 ▪ General Settings for a Web Site

Setting	Description
Feature	Select the feature with which the Web site is associated. When the selected feature is installed, the Web site and all of its applications and virtual directories are installed. If the selected feature is not installed, the Web site and its applications and virtual directories are not installed.
ASP.NET Version	To set the ASP.NET version for the Web site, enter the complete version number, or select it from the list. For example, to specify version 2 of ASP.NET, type **`2.0.50727`**. To specify version 1.1 of ASP.NET, type **`1.1.4322`**. If you specify an ASP.NET version for a Web site, IIS uses that same version number for any of the Web site's applications. ***Important ▪*** *If your installation may be run on a Windows Vista or later system, you may not want to set the ASP.NET version. Also note that if you specify version 3 of ASP.NET, an error occurs at run time. For more information, see Setting the ASP.NET Version for a Web Site or Application.*
Delete on Uninstall	Specify whether you want to delete the selected Web site during uninstallation. For more information, see Uninstalling Web Sites, Applications, and Virtual Directories.
Default Documents	Type the name of the default page on your Web site. To specify multiple pages, separate each with a comma. A Web site serves a default page whenever a browser request does not specify a document name.

Home Directory Settings

The following settings are available in the Home Directory area for a Web site in the Internet Information Services view.

Table 9-38 ▪ Home Directory Settings for a Web Site

Setting	Description
Content Source Path (Local or UNC)	This setting identifies the local path or network directory path that stores your Web site files. ● If the content for the Web site is on the target system, click the ellipsis button (...) in this setting to specify a local path. The Browse for Directory dialog box opens. This dialog box enables you to select a Windows Installer property (such as [IISROOTFOLDER]) or create a new one. By default, the files are stored in IISROOTFOLDER. ● If the content for the Web site is not on the target system, click the UNC button in this setting to specify a network location. Following is an example: \\server\share ***Tip ▪*** *Each Web site should have a unique physical path, especially if the Web site is going to be installed on a Windows Vista or later system or a Windows Server 2008 or later system. To learn more, see Run-Time Requirements for IIS Support.*
Script Source Access	Specify whether you want to allow end users to access source code if either read or write permissions are set. Source code includes scripts in ASP applications.
Read Access	Specify whether you want to provide end users with read access to the Web site.
Write Access	Specify whether you want to provide end users with write access to the Web site. This means that end users can change the Web site's properties on the target machine.
Directory Browsing	Specify whether you want to allow end users to see all the virtual directories and subdirectories below this Web site.
Log Visits	Specify whether you want to record visits to this Web site in a log file. Visits are recorded only if logging is enabled for this Web site.
Index this Resource	Specify whether you want to allow Microsoft Indexing Service to include this directory in a full-text index of your Web site. This setting applies to IIS 6 and earlier. IIS 7 ignores this setting.

Application Settings

The following settings are available in the Application Settings area for a Web site in the Internet Information Services view.

Table 9-39 ▪ Application Settings for a Web Site

Setting	Description
Application Mappings	To customize the directory's application mappings, click the ellipsis button (...) in this setting. This opens the Application Mappings dialog box, which enables you to add, edit, and delete a mapping between a file name extension and the application that processes those files.
MIME Types	To add, edit, or delete MIME types for the selected Web site, click the ellipsis button (...) in this setting. This opens the MIME Types dialog box, which enables you to add, edit, and delete a mapping between a file name extension and the corresponding content type that is served as a static file by the Web server on the target system to a browser or mail client.
Session Timeout (minutes)	Specify the number of minutes that a session can remain idle before the server terminates it automatically. If the end user does not refresh or request a page within the timeout period, the session ends. The default value is 20 minutes.
ASP Script Timeout (seconds)	Specify the length of time in seconds that .asp pages will allow a script to run before terminating and writing an event to the Windows Event Log. The minimum value for this property is 1 second; the default value is 90 seconds.

Security Settings

When you select a Web site in the Internet Information Services view, InstallShield displays several security-related settings in the Security area. The Security area lets you configure your Web server to verify the identify of users. You can authenticate users to prevent unauthorized ones from establishing a Web (HTTP) connection to restricted content. For more information, refer to the IIS documentation.

The Security area contains several categories of settings:

- Anonymous Connection
- Authenticated Access
- Secure Communication

The settings in the Anonymous Connection area are as follows.

Table 9-40 ▪ Anonymous Connection Settings in the Security Settings Area

Setting	Description
Enable Anonymous Access	Specify whether you want to allow users to establish an anonymous connection. If you do want to allow anonymous connections, also enter the appropriate Windows user account information. If you do not need the Web server to confirm the identity of end users before they can access the content, select No for this setting.
IIS Controls Anonymous Password	Specify whether you want to automatically synchronize your anonymous password settings with those set in Windows on the target system. If the password that you type for the anonymous account differs from the password that Windows has, anonymous authentication will not work. ***Note ▪*** *Password synchronization should be used only with anonymous user accounts defined on the local computer, not with anonymous accounts on remote computers.*
Anonymous User Name	If you are enabling anonymous connections, type the name of the anonymous account.
Anonymous Password	If you have selected No for the IIS Controls Anonymous Password setting, type the anonymous user account password. The password is used only within Windows. Anonymous users do not log on by using a user name and password.

The settings in the Authenticated Access area are as follows.

Table 9-41 ▪ Authenticated Access Settings in the Security Settings Area

Setting	Description
Basic authentication	Specify whether you want to enable the basic authentication method for collecting user name and password information for end users who access the Web site. ***Important ▪*** *With the basic method of authentication, user names and passwords are not encrypted when they are transmitted across the network. An unscrupulous end user who has network monitoring tools could intercept user names and passwords.*

Table 9-41 ▪ Authenticated Access Settings in the Security Settings Area (cont.)

Setting	Description
Integrated Windows Authentication	Specify whether you want to enable integrated Windows authentication. Integrated Windows authentication uses a cryptographic exchange with the end user's browser to confirm the identity of the user. When integrated Windows authentication is enabled, the Web site will use it only under the following conditions: ● Anonymous access is disabled. ● Anonymous access is denied because Windows file system permissions have been set, requiring end users to provide a Windows user name and password before establishing a connection with restricted content.
Forms Authentication	Set this option to **Yes** to enable forms authentication. ASP.NET forms-based authentication works well for sites or applications on public Web servers that receive many requests. This authentication mode lets you manage client registration and authentication at the application level, instead of relying on the authentication mechanisms provided by the operating system. ***Important ▪*** *Forms authentication sends the user name and password to the Web server as plain text. You should use Secure Sockets Layer (SSL) encryption for the Log On page and for all other pages in your application except the Home page.*

The settings in the Secure Communication area are as follows.

Table 9-42 ▪ Secure Communication Settings in the Security Settings Area

Setting	Description
SSL certificate	To specify a server certificate that should be installed on the target system, click the ellipsis button (...) in this setting, and then select the appropriate security certificate file (.cer or .pfx). As an alternative, you can select a certificate from the list if your project already contains one or more certificates. InstallShield stores the .cer file in the `Binary` table. If no certificate is configured to be installed, this setting is blank.
SSL certificate password	If the certificate that you specified has a password, enter it in this setting.

Advanced Settings

The following settings are available in the Advanced area for a Web site in the Internet Information Services view.

Table 9-43 ▪ Advanced Settings for a Web Site

Setting	Description
Custom Errors	To customize HTTP errors that are sent to clients when Web server errors occur, select the ellipsis button (...) in this setting. The Custom Errors dialog box opens, enabling you to specify the page that should be displayed for one or more HTTP errors. Administrators can use generic HTTP 1.1 errors, detailed custom error pages that IIS provides, or your own custom error pages that are you including in the installation.

Application and Virtual Directory Settings

Edition ▪ *The Express edition of InstallShield includes support for installing only one Web site per installation.*

The following editions of InstallShield support the creation of more than one Web site per installation. These editions also include support for managing IIS application pools and Web service extensions:

- *Premier*
- *Professional*

In addition, the Premier edition of InstallShield includes an IIS scanner that checks an existing IIS Web site and records data about the settings for the Web site, its virtual directories, its applications, and its application pools. You can import that IIS data into the Internet Information Services view in InstallShield Premier Edition. Once you have imported the IIS data into a project, you can use the Internet Information Services view to make changes to the IIS settings as needed.

You can add applications and virtual directories to your Web site in the Internet Information Services view. When you select an application or virtual directory in this view, many settings are displayed. The settings are organized into several main categories:

- General
- Virtual Directory
- Application Settings
- Security
- Advanced

General Settings

The following settings are available in the General area for an application or a virtual directory in the Internet Information Services view.

Table 9-44 ▪ General Settings for an Application or a Virtual Directory

Setting	Description
Name	Enter a name for the application or virtual directory.
ASP.NET Version	 ***Note ▪*** *This setting applies to applications but not virtual directories.* To set the ASP.NET version for the application, enter the complete version number, or select it from the list. For example, to specify version 2 of ASP.NET, type **`2.0.50727`**. To specify version 1.1 of ASP.NET, type **`1.1.4322`**. If you specify an ASP.NET version for a Web site, IIS uses that same version number for any of the Web site's applications. ***Important ▪*** *If your installation may be run on a Windows Vista or later system, you may not want to set the ASP.NET version. Also note that if you specify version 3 of ASP.NET, an error occurs at run time. For more information, see Setting the ASP.NET Version for a Web Site or Application.*
Default Documents	Type the name of the default page on your application or virtual directory. To specify multiple pages, separate each with a comma. An application or virtual directory serves a default page whenever a browser request does not specify a document name.

Virtual Directory Settings

The following settings are available in the Virtual Directory area for an application or a virtual directory in the Internet Information Services view.

Table 9-45 ▪ Virtual Directory Settings for an Application or a Virtual Directory

Setting	Description
Content Source Path (Local or UNC)	This setting identifies the local path or network directory path that stores the default files for your application or virtual directory. ● If the content for the application or virtual directory is on the target system, click the ellipsis button (...) in this setting to specify a local path. The Browse for Directory dialog box opens. By default, the files are stored in IISROOTFOLDER. ● If the content for the application or virtual directory is not on the target system, click the UNC button in this setting to specify a network location. Following is an example: \\server\share ***Tip ▪*** *Each application or virtual directory should have a unique physical path, especially if it is going to be installed on a Windows Vista and later system or a Windows Server 2008 and later system. To learn more, see Run-Time Requirements for IIS Support.*
Script Source Access	Specify whether you want to allow end users to access source code if either read or write permissions are set. Source code includes scripts in ASP applications.
Read Access	Specify whether you want to provide end users with read access to the application or virtual directory.
Write Access	Specify whether you want to provide end users with write access to the application or virtual directory. This means that end users can change the application's or virtual directory's properties on the target machine.
Directory Browsing	Specify whether you want to allow end users to see all the virtual directories and subdirectories below this application or virtual directory.
Log Visits	Specify whether you want to record visits to this application or virtual directory in a log file. Visits are recorded only if logging is enabled.
Index this Resource	Specify whether you want to allow Microsoft Indexing Service to include this application or virtual directory in a full-text index. This setting applies to IIS 6 and earlier. IIS 7 ignores this setting.

Application Settings

The following settings are available in the Application Settings area for an application or a virtual directory in the Internet Information Services view.

Table 9-46 ▪ Application Settings for an Application or a Virtual Directory

Setting	Description
Application Name	To associate the selected virtual directory with an application, specify the application name. ***Note ▪*** *This setting is available for virtual directories, but not for applications.* *If the virtual directory was created in InstallShield 2009 or earlier and then upgraded to the current version of InstallShield, this setting is displayed. Otherwise, this setting is not included.*
Application Mappings	To customize the directory's application mappings, click the ellipsis button (...) in this setting. This opens the Application Mappings dialog box, which enables you to add, edit, and delete a mapping between a file name extension and the application that processes those files.
MIME Types	To add, edit, or delete MIME types for the selected application or virtual directory, click the ellipsis button (...) in this setting. This opens the MIME Types dialog box, which enables you to add, edit, and delete a mapping between a file name extension and the corresponding content type that is served as a static file by the Web server on the target system to a browser or mail client.
Session Timeout (minutes)	Specify the number of minutes that a session can remain idle before the server terminates it automatically. If the end user does not refresh or request a page within the timeout period, the session ends. The default value is 20 minutes.
ASP Script Timeout (seconds)	Specify the length of time in seconds that .asp pages will allow a script to run before terminating and writing an event to the Windows Event Log. The minimum value for this property is 1 second and the default value is 90 seconds.
Execute Permissions	Specify what level of program execution is allowed for the selected application or virtual directory. Available options are: • **None**—Only static files such as HTML and image files can be accessed. • **Scripts Only**—Only scripts such as ASP scripts can be run. • **Scripts and Executables**—All file types can be accessed or run.

Table 9-46 ▪ Application Settings for an Application or a Virtual Directory (cont.)

Setting	Description
Application Protection	 ***Note ▪*** *This setting applies to applications but not virtual directories.* Specify the level of protection: ● **High**—The application is run in an isolated process that is separate from other processes. ● **Medium**—The application is run in an isolated pooled process with other applications. ● **Low**—The application is run in the same process as Web services. This setting applies to IIS 5 and earlier. Later versions ignore this setting.

Security Settings

When you select an application or a virtual directory in the Internet Information Services view, InstallShield displays several security-related settings in the Security area. The Security area lets you configure your application or virtual directory to verify the identify of users. You can authenticate users to prevent unauthorized ones from establishing a Web (HTTP) connection to restricted content. For more information, refer to the IIS documentation.

The Security area contains the following categories of settings:

- Anonymous Connection
- Authenticated Access

The settings in the Anonymous Connection area are as follows.

Table 9-47 ▪ Anonymous Connection Settings in the Security Settings Area

Setting	Description
Enable Anonymous Access	Specify whether you want to allow users to establish an anonymous connection. If you do want to allow anonymous connections, also enter the appropriate Windows user account information. If you do not need the Web server to confirm the identity of end users before they can access the content, select No for this setting.

Table 9-47 ▪ Anonymous Connection Settings in the Security Settings Area (cont.)

Setting	Description
IIS Controls Anonymous Password	Specify whether you want to automatically synchronize your anonymous password settings with those set in Windows on the target system. If the password that you type for the anonymous account differs from the password that Windows has, anonymous authentication will not work. ***Note ▪*** *Password synchronization should be used only with anonymous user accounts defined on the local computer, not with anonymous accounts on remote computers.*
Anonymous User Name	If you are enabling anonymous connections, type the name of the anonymous account.
Anonymous Password	If you have selected No for the IIS Controls Anonymous Password setting, type the anonymous user account password. The password is used only within Windows. Anonymous users do not log on by using a user name and password.

The settings in the Authenticated Access area are as follows.

Table 9-48 ▪ Authenticated Access Settings in the Security Settings Area

Setting	Description
Basic authentication	Specify whether you want to enable the basic authentication method for collecting user name and password information for end users who access the application or virtual directory. ***Important ▪*** *With the basic method of authentication, user names and passwords are not encrypted when they are transmitted across the network. An unscrupulous end user who has network monitoring tools could intercept user names and passwords.*
Integrated Windows authentication	Specify whether you want to enable integrated Windows authentication. Integrated Windows authentication uses a cryptographic exchange with the end user's browser to confirm the identity of the user. When integrated Windows authentication is enabled, the Web site will use it only under the following conditions: • Anonymous access is disabled. • Anonymous access is denied because Windows file system permissions have been set, requiring end users to provide a Windows user name and password before establishing a connection with restricted content.

Advanced Settings

The following settings are available in the Advanced area for an application or a virtual directory in the Internet Information Services view.

Table 9-49 ▪ Advanced Settings for an Application or a Virtual Directory

Setting	Description
Custom Errors	To customize HTTP errors that are sent to clients when Web server errors occur, select the ellipsis button (...) in this setting. The Custom Errors dialog box opens, enabling you to specify the page that should be displayed for one or more HTTP errors. Administrators can use generic HTTP 1.1 errors, detailed custom error pages that IIS provides, or your own custom error pages that are you including in the installation.

Component Services View

The Component Services view enables you to manage COM+ server applications and components for your installation package.

Note the following information regarding component services in InstallShield:

- Only non-system COM+ applications can be added to your project. Therefore, InstallShield displays only non-system COM+ applications under the COM+ Applications explorer in the Component Services view.
- Only the COM+ applications that are installed on the local machine are included in the Component Services view and available for you to add to your projects.

When you select a COM+ application in the Component Services view, several tabs are displayed:

- Installation
- General
- Security
- Identity
- Activation
- Queuing
- Advanced
- Dump
- Pooling/Recycling

The settings on the Installation tab are InstallShield-specific settings. The settings on the other tabs are similar to those in the Component Services administrative tool in the Control Panel. For details on each of the settings on the Installation tab, see Installation Tab. To learn about the settings on the other tabs, consult the Component Services help.

***Edition** ▪ InstallShield Premier Edition and InstallShield Professional Edition provide additional functionality in the Component Services view; these editions enable you to manage both COM+ server applications and application proxies. A COM+ application proxy consists of a subset of the attributes of the server application, and it enables remote access from a client machine to the machine where the application resides.*

Installation Tab

The Installation tab is one of the tabs that is displayed when you select a COM+ application in the Component Services view.

Table 9-50 ▪ Installation Tab Settings

Setting	Description
Features	Select the feature that should contain the selected COM+ application. To add the COM+ application to a new feature, first create a feature in the Features view, and then select its check box in this list.
Refresh the COM+ settings from the client machine at build	The Component Services view shows COM+ settings that are available in Component Services on the local machine. To refresh the COM+ settings that are displayed in your project with the settings that are available from Component Services on the local machine, select this check box. InstallShield refreshes the settings each time that you build a release.
Install user identities with roles	To install the selected COM+ application with the user identities and roles that are configured for the COM+ application on your local machine, select this check box.
Install after InstallFinalize action	If the selected COM+ application contains .NET assemblies that need to be installed to the global assembly cache (GAC), select this check box. If you select this check box, the ISComponentServiceFinalize action installs the selected COM+ application after the InstallFinalize action. Windows Installer does not commit changes made in the in-script session to the GAC until InstallFinalize.
Destination	The default destination location for COM+ applications is: `[ProgramFilesFolder]COMPlus Applications\{UID}` To install the COM+ files to a different location, select the target destination. If the destination that you want to specify is not available in the list, select the **Browse, create, or modify a directory entry** option.

***Edition** ▪ InstallShield Premier Edition and InstallShield Professional Edition provide additional functionality in the Component Services view; these editions enable you to manage both COM+ server applications and application*

proxies. A COM+ application proxy consists of a subset of the attributes of the server application, and it enables remote access from a client machine to the machine where the application resides.

Services View

***Edition** ▪ The Express edition of InstallShield includes support for installing a service during installation, and removing the service during uninstallation. It also has support for optionally starting the service after installing it, starting it automatically every time that the system starts, or starting it on demand (when the service is requested through the Service Control Manager).*

The Premier and Professional editions of InstallShield include additional flexibility for services. These editions enable you to start, stop, or delete the service during installation or uninstallation. These editions also let you configure extended service customization options that are available with Windows Installer 5. In addition, the Premier and Professional editions let you configure a service that is already present on the target system.

You can use the Services view to specify information about a service that you want to install during installation and remove during uninstallation. To add the service, first add the service executable file to your project through the Files view. Then, in the Services view, right-click the Services node and then click Add Service. Next specify the service name for the service that you are configuring. The name that you enter must match the name that is shown on the service's Properties dialog box.

***Note** ▪ A service must be a single executable file (.exe), since the Windows Installer does not support driver services.*

You must be familiar with the technical details of your service before you can configure its settings.

For details about each of the settings that you can configure for a service, see Services View Settings.

Services View Settings

***Edition** ▪ The Express edition of InstallShield includes support for installing a service during installation, and removing the service during uninstallation. It also has support for optionally starting the service after installing it, starting it automatically every time that the system starts, or starting it on demand (when the service is requested through the Service Control Manager).*

The Premier and Professional editions of InstallShield include additional flexibility for services. These editions enable you to start, stop, or delete the service during installation or uninstallation. These editions also let you configure extended service customization options that are available with Windows Installer 5. In addition, the Premier and Professional editions let you configure a service that is already present on the target system.

***Tip** ▪ The View Filter at the top of the Services view lets you select a feature whose service data you want to display in the view. The View Filter lists your project's hierarchy of features and subfeatures. Selecting a feature shows all of the services in that feature.*

When you select a service in the Services explorer of the Services view, the following settings are available for you to configure.

Install Settings

Use the Install Settings area to specify information such as the display name and description of your service, as well as when the service should be started.

Table 9-51 ▪ Install Settings

Setting	Description
Target	This setting indicates the executable file that contains the service. To modify this, click the ellipsis button (...) in this setting.
Display Name	Enter the name that you want to be displayed in the service control manager for this service. If you leave this setting blank, the service's name (that is, the text that is used for the name of your service's subnode under the Advanced Settings node) is used.
Description	Enter a description of the service. This description is registered on the target system when the service is installed, and it is displayed in the service control manager's Description column. It is also displayed in the Description box on the General tab of the service's Properties dialog box.
Service Type	Select the type of service that you are installing. Available options are: ● Win32 that runs in its own process ● Win32 that shares a process The WIN32_OWN_PROCESS type of service contains the code for only one service. The WIN32_SHARE_PROCESS type of service contains code for more than one service, enabling them to share code.
Interact with Desktop	Specify whether the service interacts with the desktop. If your service has a user interface, select Yes. If you select Yes, the User Name setting must be left blank, since the service will be installed to run in the built-in LocalSystem account.
Start Type	Specify when to start the service. Available options are: ● **Automatic**—The service starts automatically when the system starts. ● **On Demand**—The service starts when the service is requested through the service control manager. ● **Disabled**—The service cannot be started. Note that some services may support other start types (that is, during operating system initialization or by the operating system loader). However, these options are not available for the Start Type setting because the Windows Installer does not include support for them.

Table 9-51 ▪ Install Settings (cont.)

Setting	Description
Error Control	Select the appropriate action that the service control manager should perform if the service fails to start. Available options are: ● Log the error and continue ● Log the error, display a message, and continue ● Log the error and restart
Abort Install on Failure	Specify whether the entire installation should fail if the service cannot be installed on the target system. The default value is No. ***Note ▪*** *If you select Yes for this setting and end users run the installation in silent or basic UI mode, Windows Installer 3 or later must be present on the target system.*
Load Order Group	Enter the name of the load-ordering group, if any, of which this service is a member.
Dependencies	Enter any service or load-ordering groups that this service requires. The system attempts to start the required service or at least one member of the load-ordering group before starting this service. Separate multiple dependencies with a comma (,). You must precede the name of each load-ordering group with the SC_GROUP_IDENTIFIER—which is typically the plus sign (+)—so that the service control manager can distinguish it from a service.

User Name	Enter the account under which the service will be logged on. To install the service under the local system account, leave this setting blank. (Microsoft does not recommend installing services that impersonate the privileges of a single user.)
	If the service type is **Win32 that runs in its own process**, the value that you enter should use the following format:
	`DomainName\UserName`
	If the service will be logged on under the built-in domain, you can use the following format:
	`.\UserName`

Password	Enter a password for this service. Leave this setting blank if the User Name setting is empty—that is, when the service is logged on under the local system account. The password is not used if you have not specified a user name.

Table 9-51 ▪ Install Settings (cont.)

Setting	Description
Start Parameters	Enter any command-line parameters or properties that are required to run the service.
Start on Install	Specify whether you want to start the service during installation. If you select Yes, use the Arguments setting to specify the arguments that you want to be passed to the service.
Arguments	Specify the arguments that you want to be passed to the service. Separate multiple arguments with a comma (,).

Customize the Setup Appearance View

The appearance of your setup is one of the main aspects that differentiates you from your competition. You can easily customize the way your setup looks and behaves through the views listed below.

Table 9-52 ▪ Views Under the Customize the Setup Appearance View

View	Description
Dialogs	Through the Dialogs view you can select which dialogs you want displayed during your setup, and you can customize the look of these dialogs by adding custom banners and images.
Billboards	Billboards are images or Adobe Flash application files that are displayed for a specified amount of time during the file transfer portion of your installation. The billboards can be used to communicate, advertise, educate, and entertain end users. For example, billboards can present overviews on new features of the product being installed or other products from your company. Each billboard is a file that you or your company's graphics department creates for complete control over the look and feel of the file transfer.
Text and Messages	All the strings used during your setup are accessible through the Text and Messages view. Through this view you can change the text displayed on dialogs and error messages shown to your end users.

Dialogs View

Your setup's user interface is important in many ways, primarily because customer input and settings are usually handled through the user interface. If your user interface is difficult to navigate and understand, users may experience problems installing your product. In order to ease the process of creating your setup and to improve the end-user experience, InstallShield provides several predefined dialogs.

Although you are limited to the provided dialogs, you can customize many of them to produce the look and functionality you require. For example, you can add a custom image to the top of every dialog, thereby branding it with your company logo.

For information on customizing each dialog, as well as sample screen shots of each dialog in both available dialog themes, see the following:

- Splash dialog
- Install Welcome dialog
- License Agreement dialog
- Readme dialog
- Customer Information dialog
- Destination Folder dialog
- Database Folder dialog
- Setup Type dialog
- Custom Setup dialog
- Ready To Install dialog
- Setup Progress dialog
- Setup Complete Success dialog

To learn about the global settings that affect all dialogs in your project, see Global Dialog Settings for All End-User Dialogs.

Billboards View

You can add billboards to your projects to display information to end users during the installation process. The billboards can be used to communicate, advertise, educate, and entertain end users. For example, billboards can present overviews on new features of the product being installed or other products from your company. Each billboard is a file that you or your company's graphics department creates for complete control over the look and feel of the file transfer.

If you add one or more billboards to your project, the billboards are displayed at run time as the Setup Progress dialog reports the modifications that Windows Installer is making to the system. You can control the length of time the billboard is shown and its position by configuring its settings in the Billboards view.

To learn about the settings in the Billboards view, see:

- Billboard Settings—These are project-wide billboard settings.
- Settings for Adobe Flash Application File Billboards and Image Billboards—These settings are displayed in the right pane in the Billboards view when you click a Flash billboard or image billboard in the center pane.

Billboard Settings

When you click the Billboards explorer in the center pane of the Billboards view, InstallShield displays the following settings in the right pane. These are project-wide settings for billboards.

Table 9-53 ▪ Billboard Settings

Setting	Description
Billboard Type	Select the type of billboard that you want to use for your installation. Available options are: • **Fullscreen with Small progress (displayed in lower right)**—When the installation displays the standard end-user dialogs, it also displays a full-screen background. During file transfer, the installation shows full-screen backgrounds, with billboards in the foreground, and a small progress box in the lower-right corner of the screen. • **Windowed with Standard progress**—During file transfer, the installation displays a standard-size dialog that shows the billboards. The bottom of this dialog shows the progress bar. The installation does not display a background for this style. • **Windowed with Small (displayed in lower right, no billboards)**—The installation displays a small progress box in the lower-right corner of the screen during file transfer, but it does not display any billboards or a background. For more information, including sample screen shots of each billboard type, see Types of Billboards.
Loop Billboards	Specify whether you want your installation to continuously loop the image billboards until the file transfer completes and the installation shows the appropriate Setup Complete dialog. If you select No for this setting and the file transfer takes more time than you have allocated for the billboards, the installation continues displaying the last image billboard until the file transfer ends. If you select Yes for this setting and the file transfer takes more time than you have allocated for the billboards, the installation restarts the display of billboards from the beginning. The loop continues, if necessary, until the file transfer ends. The default value for this setting is No. This setting has no effect on Adobe Flash application file billboards.

Settings for Adobe Flash Application File Billboards and Image Billboards

A Flash or image billboard's settings determine which file is shown, how long it is displayed, and its position on the screen. To access these settings, open the Billboards view, and in the Billboards explorer, select the billboard that you want to configure.

Table 9-54 ▪ Settings for Adobe Flash Application File Billboards and Image Billboards

Setting	Description
File Name	Do one of the following: ● **For an Adobe Flash application file billboard**—Enter the path to the Flash application file (.swf) that you would like to use for the selected billboard, or click the ellipsis button (...) to browse to the file. Flash application files can consist of videos, movies, sounds, interactive interfaces, games, text, and more—anything that is supported by the .swf type of file. It is recommended that files such as Flash video files (.flv) and MP3 audio files be embedded in the .swf file so that they are available locally on the target system during file transfer. Although .swf files can reference external files that you can post on a Web site, this external implementation would require that end users have an Internet connection. ● **For an image billboard**—Enter the path to the image file (.bmp, .gif, .jpg, or .jpeg) that you would like to use for the selected billboard, or click the ellipsis button (...) to browse to the file. Note that animated .gif files are not supported. If you want to use animation in a billboard, consider using an Adobe Flash application file billboard. ***Note ▪*** *If the version of Flash or other tool that you use to create your .swf file is newer than the version of the Flash Player that is installed on a target system, it is possible that some of the Flash features may not work as expected on that target system.*
Duration	Enter the amount of time, in seconds, that this billboard should be displayed. The number that you enter must be from 1 to 32767 (which is a little more than 9 hours). The effect that the duration has on the run-time behavior depends on whether the installation is displaying a Flash billboard or image billboards. To learn more, see Run-Time Behavior of an Installation that Includes Billboards.

Table 9-54 ▪ Settings for Adobe Flash Application File Billboards and Image Billboards (cont.)

Setting	Description
Origin	Select where on the screen you want your billboard to be anchored. Available options are: • Upper Right • Upper Left • Lower Right • Lower Left • Centered The X and Y coordinates are measured from this point. ***Note ▪*** *This setting is used only if the* ***Fullscreen with Small progress (displayed in lower right)*** *option is selected for the Billboard Type setting.*
X Coordinate	To change the horizontal location of your billboard relative to the location you selected for the Origin setting, enter the distance, in pixels. For example, if the billboard's origin is Lower Left, an X Coordinate value of 100 places the left side of the billboard 100 pixels from the left side of the screen. ***Note ▪*** *This setting is used only if the* ***Fullscreen with Small progress (displayed in lower right)*** *option is selected for the Billboard Type setting.*
Y Coordinate	To change the vertical location of your billboard relative to the location you selected for the Origin setting, enter the distance, in pixels. For example, if the billboard's origin is Lower Left, a Y Coordinate value of 100 places the bottom of the billboard 100 pixels from the bottom of the screen. ***Note ▪*** *This setting is used only if the* ***Fullscreen with Small progress (displayed in lower right)*** *option is selected for the Billboard Type setting.*
Effect	Select the transition effect for this billboard. Rather than just appearing on the screen and disappearing after an allotted amount of time, a transition effect makes the change between billboards much smoother. ***Note ▪*** *This setting is applies to image billboards, but not to Adobe Flash application file billboards.* *In addition, this setting is used only if the* ***Windowed with Standard Progress*** *option or the* ***Fullscreen with Small progress (displayed in lower right)*** *option is selected for the Billboard Type setting.*

Table 9-54 ▪ Settings for Adobe Flash Application File Billboards and Image Billboards (cont.)

Setting	Description
Background Color	This setting displays the currently selected background color for your billboard. To change this color, click the ellipsis button (...). InstallShield displays the Color dialog box, which lets you select a predefined color or define a custom color for the background. ***Note ▪*** *This setting is used only if the* ***Fullscreen with Small progress (displayed in lower right)*** *option is selected for the Billboard Type setting.*
Title	Enter the title of this billboard, as you want it to appear in the upper-left corner of the background. ***Tip ▪*** *The maximum number of characters that the installation can display for the title at run time varies, depending on the font, the font size, the font attributes, and the length of the title string that you specify for this setting. It also depends on the screen resolution on the target system. Therefore, if you specify a long title, preview the billboard using different screen resolutions to test whether the entire title will be displayed at run time.* ***Note ▪*** *This setting is used only if the* ***Fullscreen with Small progress (displayed in lower right)*** *option is selected for the Billboard Type setting.*
Background Style	Select the style of background that you would like to use. Available options are: • **Gradient**—The background fades from dark to light. • **Solid**—The background is displayed as one solid color. ***Note ▪*** *This setting is used only if the* ***Fullscreen with Small progress (displayed in lower right)*** *option is selected for the Billboard Type setting.*
Font	Click the ellipsis (...) button to select the font that you want to use for the title in the selected billboard's background. If the target machine does not have the font you selected, a default system font is used instead. ***Note ▪*** *This setting is used only if the* ***Fullscreen with Small progress (displayed in lower right)*** *option is selected for the Billboard Type setting.*

Text and Messages View

The installation of your software is often the first contact your customers have with your application. If the setup process is hard to follow or the messages on the dialogs unclear, that first impression may be poor. Instead, you can customize all of the text that is displayed in your setup. Additionally, you can export all run-time strings to a text file, translate them, and then import the new language into the IDE for globalized versions of your setup.

The Text and Messages view lists all of the dialogs and error messages that can be displayed during the setup process. If you click one of these dialogs, you can see a picture of the dialog and all the strings that are associated with it. All run-time strings are accessible in this view.

Define Setup Requirements and Actions View

Custom actions and setup requirements are both useful ways to achieve functionality that is not supported by the Windows Installer service. Through custom actions, you can launch executable files, run script, or call functions from DLLs. By setting requirements, you can ensure that your product does not get installed onto a machine that does not meet your software's hardware needs.

Table 9-55 ▪ Views Under the Define Setup Requirements and Actions View

View	Description
Requirements	If your application has certain hardware or operating system requirements that must be met in order to run properly, you can ensure that it will not be installed onto machines that do not meet these requirements.
Custom Actions	Custom actions allow you to add functionality to your setup that is not inherently supported by Windows Installer. These actions can be used for anything from displaying a readme file to deleting registry entries.
Setup Files	If there are specific files that you need to be available to your setup project only during installation, you can add those files in the Setup Files view.

Requirements View

The Requirements view enables you to specify certain machine environments and software that your application requires in order to run properly. If these system or software requirements are not met, the installation exits. Because your application cannot be installed on systems that do not meet your hardware or software requirements, you may want to set those requirements liberally.

System Hardware Requirements

The following table describes the options available.

Table 9-56 ▪ System Requirement Conditions

Condition	Description
OS Version	To specify or change the operating system requirements, click the ellipsis button (...) in this setting. The default value of this setting is Any OS Version, which indicates that your product does not require a specific operating system. If this setting lists specific operating systems, your installation can be run on those operating systems. If the installation is run on an operating system that is not listed, the installation exits and your product is not installed.
Processor	Select the minimum required processor that your product requires.
RAM	Select the lowest amount of RAM that is required to run your product.
Screen Resolution	Select the minimum screen resolution that is required by your product.
Color Depth	Select the minimum required color depth for your product.

System Software Requirements

For information about specifying software requirements, see Specifying Software Requirements for Your Product.

Custom Actions View

As complex as Windows Installer is, it may not provide all the functionality that you require. Microsoft allows you to add flexibility to your installation that is not directly supported by Windows Installer through the use of custom actions.

InstallShield supports custom actions that call a DLL function, launch an executable file, or use a VBScript or JScript. Examples of scenarios where you may want to use custom actions include the following:

- Validating a serial number that an end user entered during installation run time
- Checking if a product is installed
- Restarting the target machine after the installation

When you add a custom action in the Custom Actions view, you need to schedule it for one of the available sequences. For example, you may want a particular custom action to be launched during the installation immediately after the Setup Complete Success dialog is displayed. In this case, you would add your custom action to the **After Setup Complete Success dialog** item in the Custom Actions view. Once you have added an action to a sequence, you can reschedule it if necessary. For more information, see Changing When Custom Actions Are Launched.

When you select a custom action in the Custom Actions view, you can configure its settings. To learn more, see the following:

- MSI DLL Custom Action Settings
- DLL Custom Action Settings
- Executable File Custom Action Settings
- VBScript Custom Action Settings
- JScript Custom Action Settings

For comparisons of each type of action, see Using Custom Actions.

MSI DLL Custom Action Settings

When you add a Windows Installer DLL (MSI DLL) custom action to your project in the Custom Actions view, you need to configure its settings.

Table 9-57 ▪ Settings for MSI DLL Custom Actions

Settings	Description
Source Location	Specify where the file that is used for this custom action is located: • **Browse File System**—The file that is used for this custom action is on your system. • **Installed with the Product**—The file will be installed on the target system. Depending on which part of the installation, uninstallation, or maintenance you schedule the custom action, this option may not be available. For example, if you schedule the custom action for the Before File Transfer part of the installation—which occurs before the file has been installed on the target system—the Installed with the Product option is not available.
File Name	If you selected Browse File System in the Source Location setting, click the ellipsis button (...) in the File Name setting to browse to the DLL file that you want to use for the custom action. When you build your installation, this file is automatically incorporated into your installation. Although the custom action is run during installation, the file is removed following installation. If you selected Installed with the Product in the Source Location setting, click the ellipsis button (...) in the File Name setting to browse for the file from the list of files that have already been added to your project through the Files view. Files that are installed on the target system are launched from their destination on the target system.

Table 9-57 ▪ Settings for MSI DLL Custom Actions (cont.)

Settings	Description
Function Name	Enter the name of the entry-point function that you want to call from your Windows Installer DLL. Note that the function must be defined with the following signature: `UINT __stdcall FunctionName (MSIHANDLE hInstall) {...}` The function name can vary; however, the return type, calling convention, and single parameter must use the types that are specified in the aforementioned signature. The MSIHANDLE data type that is used in the function signature is a handle to the running installation.
In-Script Execution	Select the iteration of the sequence that should trigger the selected custom action. For detailed information about each option, see Action Execution Options. Note that this setting does not apply to some sequences. For example, if you add a custom action under **After Initialization (before first dialog)**, the action will be scheduled for immediate execution; it cannot be scheduled for deferred, rollback, or commit execution.
Wait For Action	Specify whether you want the installation, uninstallation, or maintenance to wait until the custom action exists before proceeding with the rest of the installation. If you select No, the installation, uninstallation, or maintenance continues while your custom action runs.
Ignore Exit Code	Specify whether you want the Windows Installer to ignore the return value of this custom action and continue the installation, uninstallation, or maintenance.
Comments	Enter comments about this custom action. These comments are for your reference only and are not displayed to end users.
Condition	This setting lets you specify one or more conditions that must be true in order for the selected custom action to be run. For example, you can create a condition that checks the target system for a specific operating system or minimum system requirements. If the conditions evaluate to True at run time, the custom action is run. If one or more of the conditions evaluate as False, the custom action is not run. To specify one or more conditions, click the ellipsis button (...) in this setting. When you add a condition, InstallShield adds a new setting under the Condition setting. This new setting displays the conditional statement for the condition that you added. To edit or delete a condition, click the ellipsis button (...) in the Condition setting.

DLL Custom Action Settings

When you add a DLL custom action to the Custom Actions view, you need to configure its settings.

Table 9-58 ▪ Settings for DLL Custom Actions

Settings	Description
Source Location	Specify where the file that is used for this custom action is located: • **Browse File System**—The file that is used for this custom action is on your system. Depending on which part of the installation, uninstallation, or maintenance you schedule the custom action, this option may not be available. For example, if you schedule the custom action for the After File Transfer part of the installation, this source location option may not be applicable. • **Installed with the Product**—The file will be installed on the target system. Depending on which part of the installation, uninstallation, or maintenance you schedule the custom action, this option may not be available. For example, if you schedule the custom action for the Before File Transfer part of the installation—which occurs before the file has been installed on the target system—the Installed with the Product option is not available. The MSI DLL custom action offers more flexibility with source location options than the standard DLL custom action. For more information, see Windows Installer DLL Custom Actions.
File Name	If you selected Browse File System in the Source Location setting, click the ellipsis button (...) in the File Name setting to browse to the DLL file that you want to use for the custom action. When you build your installation, this file is automatically incorporated into your installation. Although the custom action is run during installation, the file is removed following installation. If you selected Installed with the Product in the Source Location setting, click the ellipsis button (...) in the File Name setting to browse for the file from the list of files that have already been added to your project through the Files view. Files that are installed on the target system are launched from their destination on the target system.
Function Name	Enter the name of the function you want to call from your DLL. Although your function must accept parameters, you do need not to list the parameters in this setting. For example, if your function has the following prototype, type `MyFunction` for the function name: `CHAR WINAPI MyFunction(HWND, LPSTR, LPSTR, LPSTR, LPSTR);` For more information, see Classic DLL Custom Action Function Prototype or New DLL Custom Action Function Prototype.

Table 9-58 ▪ Settings for DLL Custom Actions (cont.)

Settings	Description
Function Signature	Select the format that your DLL function uses. Available options are: • **Classic**—This format requires a precise prototype for the entry-point function. This format was used in early versions of InstallShield Express. For more information, see Classic DLL Custom Action Function Prototype. • **New**—This format enables you to get a handle to the .msi database that is currently running. Once you have a handle to the database, you can call any number of Windows Installer APIs. For more information, see New DLL Custom Action Function Prototype. Note that the prototype in this type of DLL uses the same signature that is required for MSI DLL custom actions. If you want to use this type of signature for a new custom action that you are adding to your project, it is recommended that you use an MSI DLL custom action instead of a standard DLL, since an MSI DLL action offers more flexible scheduling options. To learn more, see Windows Installer DLL Custom Actions.
In-Script Execution	Select the iteration of the sequence that should trigger the selected custom action. For detailed information about each option, see Action Execution Options. Note that this setting does not apply to some sequences. For example, if you add a custom action under **After Initialization (before first dialog)**, the action will be scheduled for immediate execution; it cannot be scheduled for deferred, rollback, or commit execution.
Wait For Action	Specify whether you want the installation, uninstallation, or maintenance to wait until the custom action exists before proceeding with the rest of the installation. If you select No, the installation, uninstallation, or maintenance continues while your custom action runs.
Ignore Exit Code	Specify whether you want the Windows Installer to ignore the return value of this custom action and continue the installation, uninstallation, or maintenance.
Comments	Enter comments about this custom action. These comments are for your reference only and are not displayed to end users.

Table 9-58 ▪ Settings for DLL Custom Actions (cont.)

Settings	Description
Condition	This setting lets you specify one or more conditions that must be true in order for the selected custom action to be run. For example, you can create a condition that checks the target system for a specific operating system or minimum system requirements. If the conditions evaluate to True at run time, the custom action is run. If one or more of the conditions evaluate as False, the custom action is not run. To specify one or more conditions, click the ellipsis button (...) in this setting. When you add a condition, InstallShield adds a new setting under the Condition setting. This new setting displays the conditional statement for the condition that you added. To edit or delete a condition, click the ellipsis button (...) in the Condition setting.

Executable File Custom Action Settings

When you add a custom action that launches an .exe file to the Custom Actions view, you need to configure its settings.

Table 9-59 ▪ Settings for Executable File Custom Actions

Setting	Description
Source Location	Specify where the file that is used for this custom action is located: ● **Browse File System**—The file that is used for this custom action is on your system. ● **Installed with the Product**—The file will be installed on the target system. Depending on which part of the installation, uninstallation, or maintenance you schedule the custom action, the Installed with the Product option may not be available. For example, if you schedule the custom action for the Before File Transfer part of the installation—which occurs before the file has been installed on the target system—the Installed with the Product option is not available. ● **File Exists on Target System**—The file already exists on the target system. ***Tip ▪*** *If you select File Exists on Target System and specify* `[SUPPORTDIR]` *for the File Location setting, you can launch a file that you added in the Setup Files view as a custom action.*

Table 9-59 ▪ Settings for Executable File Custom Actions (cont.)

Setting	Description
File Name	If you selected Browse File System in the Source Location setting, click the ellipsis button (...) in the File Name setting to browse to the .exe file that you want to use for the custom action. When you build your installation, this file is automatically incorporated into your installation. Although the custom action is run during installation, the file is removed following installation. If you selected Installed with the Product in the Source Location setting, click the ellipsis button (...) in the File Name setting to browse for the file from the list of files that have already been added to your project through the Files view. Files that are installed on the target system are launched from their destination on the target system. This setting is displayed if you selected one of the following options for the Source Location setting: • Browse File System • Installed with the Product
File Location	Select the folder on the target system that contains the executable file that you want to be launched, or click the ellipsis button (...) to select or create the folder. For example, if you want to launch `Notepad.exe`, enter **`[SystemFolder]`** in the File Location setting, and enter **`Notepad.exe`** in the File Name and Command Line setting. This setting is displayed if you selected File Exists on Target System for the Source Location setting.
Command Line	Enter any command-line parameters that you want to pass to your executable file. For example, if you select Notepad as your executable file, you can pass a file name at the command line to open the file when the executable file is launched. In this case, you might enter **`E:/Readme.txt`** in order to launch a readme file. Enclose long file names in quotation marks. This setting is displayed if you select one of the following options for the Source Location setting: • Browse File System • Installed with the Product
File Name and Command Line	Enter the name of the file. Do not add a directory before the file name—this is provided in the File Location setting. After the file name, enter any command line that you want to send to the executable file. For example, if you want to launch `Notepad.exe`, enter **`[SystemFolder]`** in the File Location setting, and enter **`Notepad.exe`** in the File Name and Command Line setting. This setting is displayed if you selected File Exists on Target System for the Source Location setting.

Table 9-59 ▪ Settings for Executable File Custom Actions (cont.)

Setting	Description
In-Script Execution	Select the iteration of the sequence that should trigger the selected custom action. For detailed information about each option, see Action Execution Options. Note that this setting does not apply to some sequences. For example, if you add a custom action under **After Initialization (before first dialog)**, the action will be scheduled for immediate execution; it cannot be scheduled for deferred, rollback, or commit execution.
Wait For Action	Specify whether you want the installation, uninstallation, or maintenance to wait until the custom action exists before proceeding with the rest of the installation. If you select No, the installation, uninstallation, or maintenance continues while your custom action runs.
Ignore Exit Code	Specify whether you want the Windows Installer to ignore the return value of this custom action and continue the installation, uninstallation, or maintenance.
Comments	Enter comments about this custom action. These comments are for your reference only and are not displayed to end users.
Condition	This setting lets you specify one or more conditions that must be true in order for the selected custom action to be run. For example, you can create a condition that checks the target system for a specific operating system or minimum system requirements. If the conditions evaluate to True at run time, the custom action is run. If one or more of the conditions evaluate as False, the custom action is not run. To specify one or more conditions, click the ellipsis button (...) in this setting. When you add a condition, InstallShield adds a new setting under the Condition setting. This new setting displays the conditional statement for the condition that you added. To edit or delete a condition, click the ellipsis button (...) in the Condition setting.

VBScript Custom Action Settings

When you add a VBScript custom action to the Custom Actions view, you need to configure its settings.

Table 9-60 ▪ Settings for VBScript Custom Actions

Setting	Description
Source Location	Specify where the file that is used for this custom action is located: • **Browse File System**—The file that is used for this custom action is on your system. • **Installed with the Product**—The file will be installed on the target system. Depending on which part of the installation, uninstallation, or maintenance you schedule the custom action, the Installed with the Product option may not be available. For example, if you schedule the custom action for the Before File Transfer part of the installation—which occurs before the file has been installed on the target system—the Installed with the Product option is not available.
File Name	If you selected Browse File System in the Source Location setting, click the ellipsis button (...) in the File Name setting to browse to the VBScript file that you want to use for the custom action. When you build your installation, this file is automatically incorporated into your installation. Although the custom action is run during installation, the file is removed following installation. If you selected Installed with the Product in the Source Location setting, click the ellipsis button (...) in the File Name setting to browse for the file from the list of files that have already been added to your project through the Files view. Files that are installed on the target system are launched from their destination on the target system.
Function Name	If you have defined a function within your .vbs file, enter the name of the function that you want to execute. This is executed after the VBScript. VBScript functions cannot take arguments from custom actions. Additionally, it is unnecessary to have a function—the custom action first executes the script, then calls the function if specified.
In-Script Execution	Select the iteration of the sequence that should trigger the selected custom action. For detailed information about each option, see Action Execution Options. Note that this setting does not apply to some sequences. For example, if you add a custom action under **After Initialization (before first dialog)**, the action will be scheduled for immediate execution; it cannot be scheduled for deferred, rollback, or commit execution.
Wait for Action	Specify whether you want the installation, uninstallation, or maintenance to wait until the custom action exists before proceeding with the rest of the installation. If you select No, the installation, uninstallation, or maintenance continues while your custom action runs.

Table 9-60 ▪ Settings for VBScript Custom Actions (cont.)

Setting	Description
Ignore Exit Code	Specify whether you want the Windows Installer to ignore the return value of this custom action and continue the installation, uninstallation, or maintenance.
Comments	Enter comments about this custom action. These comments are for your reference only and are not displayed to end users.
Condition	This setting lets you specify one or more conditions that must be true in order for the selected custom action to be run. For example, you can create a condition that checks the target system for a specific operating system or minimum system requirements. If the conditions evaluate to True at run time, the custom action is run. If one or more of the conditions evaluate as False, the custom action is not run. To specify one or more conditions, click the ellipsis button (...) in this setting. When you add a condition, InstallShield adds a new setting under the Condition setting. This new setting displays the conditional statement for the condition that you added. To edit or delete a condition, click the ellipsis button (...) in the Condition setting.

JScript Custom Action Settings

When you add a JScript custom action to the Custom Actions view, you need to configure its settings.

Table 9-61 ▪ Settings for JScript Custom Actions

Setting	Description
Source Location	Specify where the file that is used for this custom action is located: ● **Browse File System**—The file that is used for this custom action is on your system. ● **Installed with the Product**—The file will be installed on the target system. Depending on which part of the installation, uninstallation, or maintenance you schedule the custom action, the Installed with the Product option may not be available. For example, if you schedule the custom action for the Before File Transfer part of the installation—which occurs before the file has been installed on the target system—the Installed with the Product option is not available.

Table 9-61 ▪ Settings for JScript Custom Actions (cont.)

Setting	Description
File Name	If you selected Browse File System in the Source Location setting, click the ellipsis button (...) in the File Name setting to browse to the JScript file that you want to use for the custom action. When you build your installation, this file is automatically incorporated into your installation. Although the custom action is run during installation, the file is removed following installation. If you selected Installed with the Product in the Source Location setting, click the ellipsis button (...) in the File Name setting to browse for the file from the list of files that have already been added to your project through the Files view. Files that are installed on the target system are launched from their destination on the target system.
Function Name	If you defined a function within your .js file, enter the name of the function that you want to be called from your JScript. JScript functions cannot take arguments from custom actions. Note that it is not necessary to have a function—the custom action first executes the script, then calls the function if one is specified.
In-Script Execution	Select the iteration of the sequence that should trigger the selected custom action. For detailed information about each option, see Action Execution Options. Note that this setting does not apply to some sequences. For example, if you add a custom action under **After Initialization (before first dialog)**, the action will be scheduled for immediate execution; it cannot be scheduled for deferred, rollback, or commit execution.
Wait For Action	Specify whether you want the installation, uninstallation, or maintenance to wait until the custom action exists before proceeding with the rest of the installation. If you select No, the installation, uninstallation, or maintenance continues while your custom action runs.
Ignore Exit Code	Specify whether you want the Windows Installer to ignore the return value of this custom action and continue the installation, uninstallation, or maintenance.
Comments	Enter comments about this custom action. These comments are for your reference only and are not displayed to end users.

Table 9-61 ▪ Settings for JScript Custom Actions (cont.)

Setting	Description
Condition	This setting lets you specify one or more conditions that must be true in order for the selected custom action to be run. For example, you can create a condition that checks the target system for a specific operating system or minimum system requirements. If the conditions evaluate to True at run time, the custom action is run. If one or more of the conditions evaluate as False, the custom action is not run. To specify one or more conditions, click the ellipsis button (...) in this setting. When you add a condition, InstallShield adds a new setting under the Condition setting. This new setting displays the conditional statement for the condition that you added. To edit or delete a condition, click the ellipsis button (...) in the Condition setting.

Setup Files View

Setup or support files are files that are available on the target system only during your application's installation process. Setup files and subfolders are copied to a temporary directory on the target system when installation begins and are deleted when the installation is complete. The support directory (SUPPORTDIR) is a dynamic file location and might be different on every target system and even on the same system for different installation instances.

The Setup Files view also contains a Disk1 node, which allows you to add files to the root of the source media.

Prepare for Release View

The final step in creating your setup project is to build and test your installation. InstallShield provides you with many different media types to choose from, as well as the ability to test your installation from within InstallShield.

Table 9-62 ▪ Views Under the Prepare for Release View

View	Description
Releases	The Releases view enables you to configure, build, test, and distribute your distribution media for your end users.

Releases View

When you build a release, InstallShield takes all of the information from your installation project and compiles it into a Windows Installer installation package (.msi file) capable of installing your product onto any supported Windows platform.

After you have completely designed your project in InstallShield, you are ready to build a release for testing and, ultimately, distribution to your customers. The Releases view contains settings that indicate how InstallShield should build your release. To learn more about the settings in the Releases view, see the following:

- Express Settings
- Media Types
- Build Tab
- Setup.exe Tab
- Signing Tab
- .NET/J# Tab
- Internet Tab
- Events Tab

When you build a release, InstallShield takes all of the information from your project and compiles it into a package that is capable of installing your product onto any supported Windows platform.

Express Settings

The Express item in the center pane of the Releases view has the following settings that apply to all of your projects releases.

Table 9-63 ▪ Express Settings

Setting	Description
Package Code	To override the default package code that is entered in the General Information view, enter a new GUID. To have InstallShield generate a different GUID for you, click the **Generate a new GUID** button ({...}) in this setting. ***Note ▪*** *This package code is ignored if Yes is selected for the Generate Package Code setting.*
Generate Package Code	Specify whether you want InstallShield to generate a new package code every time that a release is built: • **Yes**—InstallShield generates a new package code at build time and includes it in the .msi package. • **No**—InstallShield does not generate a new package code at build time. If you enter a package code in the Package Code setting, that package code is used. If you do not specify a package code for your releases, InstallShield uses the package code that it generated when you first created the project. The package code—which is part of the Summary Information Stream—identifies a particular database. Any two .msi databases with identical package codes must have identical contents. Therefore, the package code should be changed for each build.

Table 9-63 ▪ Express Settings (cont.)

Setting	Description
MSI Package File Name	Specify the file name—without the period or the file extension—that InstallShield should use for the .msi file that it generates at build time. If this setting is blank, the product name is used. You can also include the value of a property from the Property Table in this field, as described in Setup File Name.

Setup File Name	Specify the file name—without the .exe file extension—that InstallShield should use for the setup launcher file that it generates at build time. If this setting is blank, InstallShield uses the default value of *Setup*, and the setup launcher file is called `Setup.exe`.
	
	You can also include the value of a property from the Property Table in this field. For example, you could enter any of the following properties:
	
	`setup[ProductVersion]`
	`setup[CustomVersion]`
	`setup[ProductCode]`
	`setup[ProductCode][ProductVersion]`
	
	If you entered **`setup[ProductVersion]`**, it would result in a setup named `setup14.10.1234.exe`, for example.

64-Bit Setup Launcher	Specify whether you want to build a 64-Bit Setup Launcher or a 32-Bit Setup Launcher. Select `Yes` to build the 64-Bit setup launcher. Select `No` to build the 32-Bit setup launcher. For more detail, see Building 64-Bit Setup Launcher.

Media Types

Media types, or release types, represent the type of disk or other electronic medium that you use to distribute your installation. The most common of these release types is CD-ROM, although all of the following types are available:

Table 9-64 ▪ Release Types

Release Type	Description
CD-ROM	If you are distributing your installation on a CD, select this option. The maximum size for this media type is 650 MB per CD.
Custom	If the medium that you are distributing is not among the media types listed, select this option. For example, you may be distributing your installation on Zip disks.
DVD-5	If you are distributing your installation on a 4.38 GB DVD, select this option.

Table 9-64 ▪ Release Types (cont.)

Release Type	Description
DVD-9	If you are distributing your installation on a 7.95 GB DVD, select this option.
DVD-10	If you are distributing your installation on an 8.75 GB DVD, select this option.
DVD-18	If you are distributing your installation on a 15.83 GB DVD, select this option.
SingleImage	If your installation is going to be placed on a network, select this option. There is no size limit for this media type.
WebDeployment	If you are creating an installation that is designed to be deployed over the Web, you may want to select this option. There is no size limit for this media type. WebDeployment setups have specific settings associated with them.

If your installation requires more than one disk, InstallShield automatically splits it across as many disks as needed.

Caution ▪ *If you build a multiple-disk installation, you need to set the volume label on the second and subsequent disks.*

Build Tab

The Build tab is where you configure how InstallShield should package your release.

Table 9-65 ▪ Settings on the Build Tab

Setting	Media Type	Description
Release Location	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Enter the path to the top-level directory where your release will start to be built, or click the ellipsis button (...) to browse to the location.
Media Size	**Custom, CD-ROM, DVD**	This setting shows the size of the currently selected media; it is read-only for all media except custom. The decimal separator is a period (.). When you build your release, InstallShield splits the files across disk image folders that are no larger than the value that is specified in this setting. The Media Size Unit setting indicates whether the value in the Media Size setting is in megabytes or gigabytes.

Table 9-65 ▪ Settings on the Build Tab (cont.)

Setting	Media Type	Description
Media Size Unit	**Custom, CD-ROM, DVD**	This setting displays the unit size for the current media. For example, CDs are measured in megabytes and DVDs are measured in gigabytes. You cannot change this setting unless you are using the custom media type.
Cluster Size	**Custom, CD-ROM, DVD**	A cluster is the smallest unit of space on a disk. This size, measured in bytes, is the minimum amount of space that is allocated for each file. If the cluster size is set at 1024 bytes and you have a file that is 2000 bytes, that file will take up two clusters on the disk. If the file is only 1000 bytes, it requires only one cluster. However, the rest of the space in that cluster is not available for any other file. You cannot modify the cluster size for a predefined media type. This setting is read-only unless you are configuring the custom media type.
Compression	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify whether your product's data files should be compressed: ● **Compressed**—InstallShield compresses all of your product's data files into .cab files. ● **Uncompressed**—InstallShield does not compress your product's files into .cab files. ***Note ▪*** *The output of the build process depends on the media type that you are building (such as CD-ROM, DVD-5, or SingleImage), whether compression is used (as specified in the Compression setting), and whether you are including a setup launcher (as specified in the Setup Launcher setting on the Setup.exe tab).* *For example, if you select Compressed for the Compression setting, you select Yes for the Setup Launcher setting, and you are building a SingleImage release, your data files are compressed into .cab files, and the .cab files are streamed into your* `Setup.exe` *file. If you select Uncompressed for the Compression setting, you select No for the Setup Launcher setting, and you are building a CD-ROM release, your data files are left uncompressed in a subfolder of the folder that contains the .msi file.*

Table 9-65 ▪ Settings on the Build Tab (cont.)

Setting	Media Type	Description
Cab Optimization Type	Custom, CD-ROM, DVD, SingleImage, WebDeployment	If Compressed is selected for the Compression setting, specify the type of compression that InstallShield should use when building this release's .cab files. Available options are: ● **LZX**—InstallShield uses LZX compression to compress your product's data files into .cab files. This option results in the smallest .cab files; however, it also takes the most time to extract the data from these .cab files at run time. ● **MSZIP**—InstallShield uses MSZIP compression to compress your product's data files into .cab files. This is the default option. ● **None**—InstallShield does not use any compression when creating the .cab files. ***Important ▪*** *Using compression generally decreases the size of your compressed files, but the build process may take more time to complete. Depending on the number and size of the files being compressed, the LZX compression and the build may take hours to complete. Therefore, if you select the LZX option, it is recommended that your build machine have the latest hardware to minimize the time that it takes for the build to complete.*
Generate Autorun.inf	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Select Yes if you are distributing your installation on a CD-ROM or DVD-ROM and want to support the AutoPlay feature. InstallShield creates a text file called `Autorun.inf`, which contains the instructions to autoplay your installation, in the root of your disk images folder. You can edit this file to add additional AutoPlay options or to pass command-line parameters to `MsiExec.exe`, `Setup.exe`, or `Update.exe`.

Table 9-65 ▪ Settings on the Build Tab (cont.)

Setting	Media Type	Description
Keep Unused Directories	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify whether you want InstallShield to remove unused directories from the `Directory` table of the .msi file when you build this release. Available options are: • **No**—If a directory that is listed in the Directory column of the `Directory` table is not referenced in any known location in the .msi file, InstallShield removes it from the `Directory` table of the .msi file that it creates at build time. This occurs after any merge modules are merged, but only directories that are present in the .msi file are removed; therefore, if a merge module contains new unused directories in its `Directory` table, the new unused directories are added to the installation. • **Yes**—InstallShield does not remove any directories from the `Directory` table of the .msi file that it creates at build time. The default value is No. ***Note ▪*** *Under some conditions, predefined directories cannot be resolved, causing an installation to fail. Removing unused directories from the* `Directory` *table enables you to avoid unnecessary failures. Therefore, it is recommended that you select No for this setting.*

Setup.exe Tab

Note ▪ *For information on including Windows Installer redistributables in installations, see Adding Windows Installer Redistributables to Projects.*

The Setup.exe tab is where you configure settings about your `Setup.exe` file. It is also where you specify whether you want to include redistributables for Windows Installer 3.1 or earlier. For more information on Windows Installer redistributables, see Adding Windows Installer Redistributables to Projects.

Table 9-66 ▪ Settings on the Setup.exe Tab

Setting	Media Type	Description
Setup Launcher	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify whether you want to create a `Setup.exe` setup launcher. To learn about scenarios that require a setup launcher, see Creating a Setup Launcher.

Table 9-66 ▪ Settings on the Setup.exe Tab (cont.)

Setting	Media Type	Description
Required Execution Level	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Use the Required Execution Level setting to specify the minimum level required by your installation's `Setup.exe` file for running the installation (the setup launcher, any InstallShield prerequisites, and the .msi file) on Windows Vista and later platforms. The available options are: • **Administrator**—`Setup.exe` requires administrative privileges to run. Administrators must authorize it; non-administrators must authenticate as an administrator. • **Highest available**—`Setup.exe` prefers administrative privileges. Administrators must authorize it; non-administrators run it without administrative privileges. • **Invoker**—`Setup.exe` does not require administrative privileges, and all users can run it without administrative privileges. `Setup.exe` does not display any UAC messages prompting for credentials or for consent. This is the default option. InstallShield embeds a Windows application manifest in the `Setup.exe` launcher. This manifest specifies the selected execution level. Operating systems earlier than Windows Vista ignore the required execution level. The benefit of elevating the required execution level is that privileges can be elevated only once if necessary to run `Setup.exe`, and that these privileges can be carried over to all of the installation's prerequisites and the .msi file without requiring multiple prompts for approval. Thus, if two of your prerequisites require administrative privileges, for example, you can change this setting to Administrator, and then end users are prompted only once during the installation, before Windows Installer runs the Setup.exe file. Note, however, that if you elevate the privileges and also launch the application at the end of the installation, the elevated privileges are carried over to the application. In most cases, running an application with elevated privileges on Windows Vista and later platforms is discouraged. For more information, see Minimizing the Number of User Account Control Prompts During Installation.
Include MSI Engine	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify whether you want to include redistributables for Windows Installer 3.1. ***Note ▪*** *This setting applies to redistributables for Windows Installer 3.1. For information on different methods of adding various versions of Windows Installer redistributables to a project, see Adding Windows Installer Redistributables to Projects.*

Table 9-66 ▪ Settings on the Setup.exe Tab (cont.)

Setting	Media Type	Description
MSI 3.1 Engine Location	Custom, CD-ROM, DVD, SingleImage, WebDeployment	 ***Note ▪*** *This setting applies to redistributables for Windows Installer 3.1. For information on different methods of adding various versions of Windows Installer redistributables to a project, see Adding Windows Installer Redistributables to Projects.* Specify where the Windows Installer engine installers should be located. Available options are: • **Copy from Source Media**—Leave the selected Windows Installer engine installers on the root of the source media. This option is not available for WebDeployment or SingleImage media types with all of your files compressed into `Setup.exe`. • **Extract Engine from Setup.exe**—Compress the selected Windows Installer engine installers into `Setup.exe`, to be extracted at run time. • **Download Engine from the Web**—Download the Windows Installer engine installer (if necessary) from a URL that you specify. If you select this setting, be sure to set the engine URL settings to the appropriate Web locations, or leave the default value.
MSI 3.1 Engine URL	Custom, CD-ROM, DVD, SingleImage, WebDeployment	This setting specifies the location from which the setup launcher downloads the Windows Installer 3.1 engine, if needed. This property is ignored unless the MSI 3.1 Engine Location setting for the current release is set to Download Engine from the Web. Do not specify the file name in the URL. ***Note ▪*** *The default URL is a site maintained by Revenera for your convenience. The 3.1 engine is available for download from this location.*

Table 9-66 ▪ Settings on the Setup.exe Tab (cont.)

Setting	Media Type	Description
Delay MSI Engine Reboot	Custom, CD-ROM, DVD, SingleImage, WebDeployment	 ***Note ▪*** *This setting applies to redistributables for Windows Installer 3.1 and earlier. For information on different methods of adding various versions of Windows Installer redistributables to a project, see Adding Windows Installer Redistributables to Projects.* Specify whether you want to postpone any reboot associated with installing or updating the Windows Installer engine on the target system until after your installation has completed. Select Yes to postpone the reboot, if one is necessary. Select No to allow the system to reboot, if necessary, immediately after the Windows Installer engine has been installed or updated and before performing your installation.
Cache MSI Locally	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify whether the .msi file and other installation files for the current build should be cached on the target system. • **Yes**—Cache the .msi file and other installation files on the target system for use with application maintenance and repair. The Cache Path setting for the current release specifies where files should be cached. • **No**—Do not cache the .msi file and other installation files on the target system. ***Note ▪*** *This setting is enabled for releases that do not have the .msi file available in the same folder as the* `Setup.exe` *file on the target system.*
Cache Path	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify where the cached .msi file and other cached installation files should be stored on the end users's system. You can enter a hard-coded value such as **C:\CachedFiles**, but it is recommended that you cache files in a path using a destination variable selected from the list. The default value is **[LocalAppDataFolder]Downloaded Installations**. ***Note ▪*** *This setting is used only if the Cache MSI Locally setting for the current release is set to Yes.*

Table 9-66 ▪ Settings on the Setup.exe Tab (cont.)

Setting	Media Type	Description
Minimum Initialization Time	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify the minimum number of seconds that the installation should display the initialization dialog when end users run this release. InstallShield uses the value that you specify for this setting as the value of the SplashTime keyname in the `Setup.ini` file. When the installation initializes, by default, an initialization dialog is displayed. The splash screen, if provided, is also shown at this time. If you specify a minimum initialization time in this setting, the initialization dialog and splash screen are shown for at least the specified number of seconds. If initialization takes longer than the time specified, the dialog and splash screen are still displayed until the installation completes initialization. If initialization takes less time than the time specified, the installation continues to display the dialog and splash screen until the specified time has elapsed, and then the installation continues.
Password Protect Launcher	Custom, CD-ROM, DVD, SingleImage, WebDeployment	To password-protect your installation, select Yes, and then type a password for the Launcher Password setting. This setting is available only if you select Yes for the Setup Launcher setting.
Launcher Password	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Type a password to protect your installation. Passwords are case-sensitive. This setting is used only if you select Yes for the Password Protect Launcher setting.
Use Custom Version Properties	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify whether you want to override the default copyright notice and file description for `Setup.exe` with your own copyright notice and file description. If you select Yes, enter your own information in the Launcher Copyright setting and the File Description setting. The copyright and description are displayed on the Properties dialog box for the setup launcher; this Properties dialog box opens when end users right-click the `Setup.exe` file and then click Properties. To learn more, see Customizing File Properties for the Setup Launcher.

Table 9-66 ▪ Settings on the Setup.exe Tab (cont.)

Setting	Media Type	Description
Launcher Copyright	Custom, CD-ROM, DVD, SingleImage, WebDeployment	If you want to override the default copyright notice for `Setup.exe` with your product's copyright notice, enter your product's copyright notice. Note that you must also select Yes in the Use Custom Version Properties setting. The copyright is displayed on the Properties dialog box for the setup launcher; this Properties dialog box opens when end users right-click the `Setup.exe` file and then click Properties. To learn more, see Customizing File Properties for the Setup Launcher.
File Description	Custom, CD-ROM, DVD, SingleImage, WebDeployment	If you want to override the default file description for `Setup.exe` with your own description, enter the appropriate description. Note that you must also select Yes in the Use Custom Version Properties setting. The description is displayed on the Properties dialog box for the setup launcher; this Properties dialog box opens when end users right-click the `Setup.exe` file and then click Properties. To learn more, see Customizing File Properties for the Setup Launcher.
File Version	Custom, CD-ROM, DVD, SingleImage, WebDeployment	If you want to override the default file version for `Setup.exe` with your own file version, enter the appropriate file version number. Note that you must also select Yes in the Use Custom Version Properties setting. The file version is displayed on the Properties dialog box for the setup launcher; this Properties dialog box opens when end users right-click the `Setup.exe` file and then click Properties. To learn more, see Customizing File Properties for the Setup Launcher.

Table 9-66 ▪ Settings on the Setup.exe Tab (cont.)

Setting	Media Type	Description
InstallShield Prerequisites Location	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify where the InstallShield prerequisites that are selected in the Redistributables view should be located. Available options are: • **Follow Individual Selections**—Use the locations that are specified for each individual InstallShield prerequisite's properties in the Redistributables view. • **Download From The Web**—Download all of the InstallShield prerequisite files included in your project (if necessary) from the URL specified in the InstallShield prerequisite (.prq) file for each prerequisite. This option overrides the locations that are specified in the Redistributables view for each InstallShield prerequisite's properties. This option is recommended if your installation will be downloaded from the Internet and you want to minimize the package size and download time. An InstallShield prerequisite will not be downloaded if the correct version is already present on the target machine. • **Extract From Setup.exe**—Compress the InstallShield prerequisite files into Setup.exe, to be extracted at run time, if necessary. This option overrides the locations that are specified in the Redistributables view for each InstallShield prerequisite's properties. Select this option if the entire installation must be self-contained in `Setup.exe`. Note that the Download From The Web option results in smaller installations and shorter download time; however, the Extract From Setup.exe option provides for a completely self-contained installation. • **Copy From Source Media**—Store the InstallShield prerequisite files on the source media. This option overrides the locations that are specified in the Redistributables view for each InstallShield prerequisite's properties. Note that if an InstallShield prerequisite is added to a project as a dependency of another prerequisite, the location for the prerequisite dependency follows the location setting of the prerequisite that requires it.

Table 9-66 ▪ Settings on the Setup.exe Tab (cont.)

Setting	Media Type	Description
		 Tip ▪ *If you select the Extract From Setup.exe option or the Copy From Source Media option and then build a release that includes an InstallShield prerequisite that is not available on your computer, one or more build errors are generated for every file that the prerequisite requires. To avoid these build errors, use the Redistributables view to either download the InstallShield prerequisite from the Internet to your computer or remove it from your project before building the release.* To learn more, see Specifying the Run-Time Location for InstallShield Prerequisites at the Release Level.

Signing Tab

The Signing tab is where you specify the digital signature information—including the digital certificate files that a certification authority grated to you—that InstallShield should use to sign your files. It is also where you specify which files in your installation should be digitally signed at build time.

Table 9-67 ▪ Settings on the Signing Tab

Setting	Media Type	Description
Signing Type	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify the method to digitally sign build-generated files. Specify the argument for a sign tool's configuration: • **Standard**—Select this option to use the default InstallShield Express Edition sign tool to digitally sign build-generated files. • **Custom**—Select this option to use a customized sign tool to digitally sign build-generated files. Selecting this option enables the **Path** and **Arguments** settings. ***Note ▪*** *By default, this setting is set to* ***Standard****.*
Path	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify the sign tool's location to digitally sign build-generated files by using that sign tool. To specify sign tool's location, click the ellipsis button (...) in this setting.

Table 9-67 ▪ Settings on the Signing Tab (cont.)

Setting	Media Type	Description
Arguments	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify the command-line argument for a sign tool's configuration to digitally sign build-generated files. For example, command-line argument below can be used if the Microsoft built-in signing tool is configured as a custom option to sign the binaries: `sign /fd SHA256 /f "<ProgramFilesFolder>\testCA.pfx" /t http://timestamp.digicert.com /p MyPassword [filename]` The `[filename]` variable is a place holder for full file path to be signed. It resolves to full path of the binary file to be signed during build time. By default, a file path will be added at the end of an argument and passed to a custom sign tool. Instead of using a hard-coded path, you can use the path variables, environment variables, or property names that are defined in your project. ***Note ▪*** *To utilize a property name in the* ***Arguments*** *setting, it must be specified within square brackets ([]).*
Certificate URL	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Type a fully qualified URL—for example, **http://www.mydomain.com**. This URL is used in your digital signature to link to a site that you would like end users to visit to learn more about your product, organization, or company.

Table 9-67 ▪ Settings on the Signing Tab (cont.)

Setting	Media Type	Description
Digital Certificate Information	Custom, CD-ROM, DVD, SingleImage, WebDeployment	To specify the digital certificate that you want to use to sign your release, click the ellipsis button (...) in this setting. The Certificate Selection dialog box opens, enabling you to specify either the location of the .pfx file, certificate file (EV exported .cer file), or information about the certificate store that contains the certificate. InstallShield provides an option to encrypt and store an EV token password in the project file using the public key certificate (.cer) file. The .cer file is generally created by exporting a public key from the EV Authentication Client tools associated with a USB eToken provider (for example, SafeNet Authentication Client). InstallShield displays additional options to configure the Extended Validation (EV) certificate properties if the .cer file is specified. For more details, see ● Configuring Extended Validation (EV) Certificate information in InstallShield ● Certificate Selection Dialog Box After specifying the .pfx file or choosing the certificate from test store, the below will be displayed: ● **Certificate Thumbprint**—This read-only setting displays the certificate thumbprint. ● **Issued By**—This read-only setting displays the certificate issuer information. ● **Expiration Date**—This read-only setting displays the certificate's expiration date. After specifying the certificate file (EV exported .cer file), the below settings will be displayed: ● **Cryptographic Provider**—This setting allows you to specify the cryptographic service provider (CSP). ● **Container Name**—This setting allows you to specify the private key container name associated to the cryptographic service provider (CSP). ● **Token Password**—This setting allows you to specify the EV token password which is encrypted and stored in a project file. ***Note ▪*** *Note the following informations while utilizing the settings that appear after specifying the certificate file (EV exported .cer file):* ● *Both the Cryptographic Service Provider (CSP) name and Container name can be obtained from the Private Key Certificate properties of a user certificate in the EV Vendor Authentication Client tool.* ● *An EV certificate vendor determines an EV token password's expiration period and number of invalid password attempts before it is locked. Therefore, selecting this setting requires changing your password in specific intervals.*

Table 9-67 ▪ Settings on the Signing Tab (cont.)

Setting	Media Type	Description
(Continued)	(Continued)	• *If an EV token password is locked, unlocking/resetting the token password requires an administrator password.*
Certificate Password	Custom, CD-ROM, DVD, SingleImage, WebDeployment	If the .pfx file that you are using has a password, enter it. InstallShield encrypts the password and stores it in your project file (.ise). At build time, InstallShield uses the password to sign files with a .pfx file. If your certificate is protected by a password but you do not enter it in this setting, signing with a .pfx file fails. Note that if you configure your project to use a certificate that was imported with password protection into a store, Windows prompts for the password at build time when InstallShield is attempting to sign your project's files. The strong key protection that Windows uses does not permit InstallShield to provide the password to the cryptographic provider.
Sign Output Files	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify which files you want to be signed. Available options are: • **None**—To avoid signing your installation, select this option. • **Setup.exe**—To sign your `Setup.exe` file, select this option. • **Setup.exe and Windows Installer Package**—To sign your `Setup.exe` file and your Windows Installer package (.msi), select this option. • **Windows Installer Package**—To sign your Windows Installer package (.msi), select this option.
Use 64-Bit Signing	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify whether you want to use the 64-bit signing framework to digitally sign your package. Available options are: • **Yes**—Enables the 64-bit signing framework to digitally sign a package. • **No**—Enables the 32-bit signing framework to digitally sign a package. By default, this setting is set to **No**.
Signature Description	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify the signature description that you want to use for files that are specified in the Sign Output Files setting. The description that you specify is displayed on the User Account Control (UAC) box to the right of the "Program Name:" label. The UAC dialog box opens when an end user launches the signed file and elevated privileges are required. If you leave this setting blank, InstallShield uses the name of the file without its extension as the description to the right of the "Program Name:" label on the UAC dialog box. Note that if you use the Sign Files in Package setting and its subsettings to sign the files in your package, InstallShield does not use this signature description for the UAC dialog box of the files in your package that are signed at build time.

Table 9-67 ▪ Settings on the Signing Tab (cont.)

Setting	Media Type	Description
Sign Files in Package	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify whether you want to digitally sign all the files, including InstallShield support files (English, language-independent, and advanced files) configured in the **Support Files** view, which are used only during the installation process. If you select **Yes**, use the **Include Patterns and Files** and **Exclude Patterns and Files** settings to indicate which files should be signed. ***Windows Logo ▪*** *All executable files (including .exe, .dll, .ocx, .sys, .cpl, .drv, and .scr files) in an installation must be digitally signed for the Windows logo program.*
Sign Files That Are Already Signed	Custom, CD-ROM, DVD, SingleImage, WebDeployment	If any of the files in your project are already digitally signed, determine whether you want InstallShield to replace those existing digital signatures with the digital signature that you specify on the Signing tab. Note that this affects only files that meet the requirements that are specified in the Include Patterns and Files setting and the Exclude Patterns and Files setting. ● To use the digital signature information that you are providing on the Signing tab to sign a file instead of any existing digital signature information that is already included with the file, select Yes. ● To leave the existing digital signature information intact for any files that are already signed, select No. The default value is No.
Sign Files in Their Original Location	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Determine whether you want InstallShield to sign your original files or just the files that are built into the release: ● If you want InstallShield to sign a temporary copy of each file and then use that signed temporary copy to build a release, select No. Note that if you select No, InstallShield will not modify or sign your original files. ● If you want InstallShield to sign your original files, select Yes. The default value is No.
Include Patterns and Files	Custom, CD-ROM, DVD, SingleImage, WebDeployment	To specify the files and file patterns that you want to be digitally signed at build time, do one of the following: ● To select one or more file names or file patterns from a list of all of the static files that are currently in your project, as well as file patterns such as *.dll, click the ellipsis button (...) in this setting. The Browse for file dialog box opens, enabling you to select one or more patterns and files. When you are done selecting items, InstallShield adds one or more new Include settings under the Include Patterns and Files setting. ● To type a file name or pattern manually, click the Add button in this setting. InstallShield adds a new Include setting under the Include Patterns and Files setting; use this new setting to specify the file name or pattern.

Table 9-67 ▪ Settings on the Signing Tab (cont.)

Setting	Media Type	Description
Include	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify the file or file pattern that you want to be digitally signed at build time. Note the following guidelines: ● To indicate a wild-card character, use an asterisk (*). For example, if you want to sign all .exe files, specify the following: ***.exe** Using wild-card characters is especially helpful if you include dynamically linked files in your project and you want to sign all files that match a certain pattern. ● Note that the files and file patterns that should not be signed override any files and file patterns that should be signed. For example, if you specify ***.exe** in an Include setting and in an Exclude setting, InstallShield does not sign any .exe files. To delete the file or file pattern, click the Delete button in this setting. To add another file or file pattern, use the Include Patterns and Files setting.
Exclude Patterns and Files	Custom, CD-ROM, DVD, SingleImage, WebDeployment	To specify the files and file patterns that you do not want to be digitally signed at build time, do one of the following: ● To select one or more file names or file patterns from a list of all of the static files that are currently in your project, as well as file patterns such as *.dll, click the ellipsis button (...) in this setting. The Browse for file dialog box opens, enabling you to select one or more patterns and files. When you are done selecting items, InstallShield adds one or more new Exclude settings under the Exclude Patterns and Files setting. ● To type a file name or pattern manually, click the Add button in this setting. InstallShield adds a new Exclude setting under the Exclude Patterns and Files setting; use this new setting to specify the file name or pattern.

Table 9-67 ▪ Settings on the Signing Tab (cont.)

Setting	Media Type	Description
Exclude	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify the file or file pattern that you do not want to be digitally signed at build time. Note the following guidelines: ● To indicate a wild-card character, use an asterisk (*). For example, if you do not want to sign any .drv files, specify the following: ***.drv** Using wild-card characters is especially helpful if you include dynamically linked files in your project and you want to avoid signing all files that match a certain pattern. ● Note that the files and file patterns that should not be signed override any files and file patterns that should be signed. For example, if you specify ***.exe** in an Include setting and in an Exclude setting, InstallShield does not sign any .exe files. To delete the file or file pattern, click the Delete button in this setting. To add another file or file pattern, use the Exclude Patterns and Files setting.

.NET/J# Tab

The .NET/J# tab is where you add support for the 32-bit versions of the .NET Framework 1.0, 1.1, or 2.0. It is also where you add J# support to your project.

Note ▪ *To include other versions of the .NET Framework redistributables in your project, use the Redistributables view to add the appropriate InstallShield prerequisite for the Microsoft .NET Framework to your project.*

For more information, see Adding .NET Framework Redistributables to Projects.

Table 9-68 ▪ Settings on the .NET/J# Tab

Setting	Media Type	Description
.NET Framework Location	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify where the .NET Framework runtime should be located. The .NET Framework is required for applications that are using any .NET features. Valid options are: • **Copy From Source Media**—Leave the .NET Framework runtime on the root of the source media. This option is not applicable to WebDeployment or SingleImage media types with all of your files compressed into `Setup.exe`. • **Extract From Setup.exe**—Compress the .NET Framework runtime into `Setup.exe`, to be extracted at run time. • **Download from the Web**—Download the .NET Framework runtime (if necessary) from a URL that you specify. If you select this option, you must set the value of the .NET and J# Framework URL setting to the appropriate Web location. • **Do Not Include**—Do not include the redistributable for the .NET Framework 1.0, 1.1, or 2.0 in the selected release.
.NET Framework Version	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Select the version of the .NET Framework that you want to distribute with your installation. ***Note ▪*** *To include .NET Framework 3.5, 3.0 SP1, 3.0, or 2.0 SP1 redistributables in your project, use the Redistributables view to add the appropriate InstallShield prerequisite for the Microsoft .NET Framework to your project.* *For more information, see Adding .NET Framework Redistributables to Projects.*
.NET 1.1/2.0 Core Language	Custom, CD-ROM, DVD, SingleImage, WebDeployment	If you selected .NET version 1.1 for the .NET Framework Version setting, you can specify one .NET core language that you want to distribute. This is the language that is used while the .NET 1.1 core redistributable is installed. If you selected version 2.0, the language options are all selected and disabled since they are all included with this version of the redistributable. To change the core language, click the ellipsis button (...) in this setting.

Table 9-68 ▪ Settings on the .NET/J# Tab (cont.)

Setting	Media Type	Description
Command Line to Pass to Dotnetfx.exe	Custom, CD-ROM, DVD, SingleImage, WebDeployment	 ***Note ▪*** *This setting applies only if .NET 1.1 is selected for the .NET Framework Version setting.* Enter the command line that you want to pass to Microsoft's `DotNetFx.exe`.
.NET 1.1/2.0 Language Packs	Custom, CD-ROM, DVD, SingleImage, WebDeployment	To include .NET language packs, click the ellipsis button (...) in this setting. The options that are available depend on the Microsoft language packs that you have installed on your build machine. If you click the ellipsis button, you can download additional available language packs.
Command Line to Pass to Language Packs	Custom, CD-ROM, DVD, SingleImage, WebDeployment	 ***Note ▪*** *This setting applies only if .NET 1.1 is selected for the .NET Framework Version setting.* Enter the command line that you want to send to all of the Microsoft `LangPack.exe` files that are included with the installation.
Display .NET Option Dialog	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Indicates whether `Setup.exe` displays a Yes/No message box asking the end user if they want to install .NET Framework on the target system. This setting does not determine whether your installation includes the .NET Framework, only whether the end user has a choice to install it. • **No**—Installs the .NET Framework, if required, without displaying a message box to the end user. • **Yes**—Displays the .NET option message box, which allows the end user to specify whether to install the .NET Framework. If you select Yes, a message box is displayed on the target system at run time. The message box states that the installation optionally uses the Microsoft .NET Framework and asks if they would like to install it.
Show Full User Interface when Installing .NET Framework	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify whether you want the full interface for the .NET Framework installation to be displayed. If you select Yes, the Microsoft .NET Framework (English) Setup wizard appears when `dotnetfx.exe` installs the .NET Framework on the target system. This wizard shows the progress of the .NET Framework installation. If you select No, the InstallShield Wizard appears when `dotnetfx.exe` installs the .NET Framework on the target system. This InstallShield Wizard shows the progress of the .NET Framework installation.

Table 9-68 ▪ Settings on the .NET/J# Tab (cont.)

Setting	Media Type	Description
.NET Build Configuration	Custom, CD-ROM, DVD, SingleImage, WebDeployment	This setting contains the name of the build configuration of the .NET solution. InstallShield uses this setting to determine the location of a project outputs' files (for example, Debug or Release). The C# and VB.NET Project wizards automatically populate this field when you create a new project or change the solution for the project.
.NET and J# Framework URL	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify the location from which your installation downloads the .NET Framework runtime and the J# redistributable, if included. It is not necessary to specify the file name in the URL. This setting is required only if the .NET Framework Location setting or the J# Redistributable Location setting (if included) for the current release is set to Download from the Web. The installation downloads and runs the InstallShield file `Dotnetfx.exe`, which in turn downloads `Dotnetredist.exe` from the Microsoft Web site. This behavior is hard-coded in `Dotnetfx.exe`, which can be hosted anywhere.
J# Redistributable Location	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify where the J# redistributable should be located. The .NET Framework is required for applications that use .NET features. Available options are: ● **Copy from Source Media**—Leave the J# redistributable on the root of the source media. This option is not applicable to WebDeployment or SingleImage media types with all of your files compressed into `Setup.exe`. ● **Extract from Setup.exe**—Compress the J# redistributable into `Setup.exe`, to be extracted at run time. ● **Download from the Web**—Download the J# redistributable (if necessary) from the URL specified in the .NET and J# Framework URL setting. ● **Do Not Include**—Do not include the J# redistributable in the selected release. ***Note ▪*** *The J# version number that is used matches whatever version number that you select for the .NET Framework. You cannot install version 1.1 of one of these redistributables and version 2.0 of the other.*
Command Line to Pass to the J# Redistributable	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify a command-line parameter to pass to `vjredist.exe`. Consult Microsoft Support for valid command-line parameters.

Table 9-68 ▪ Settings on the .NET/J# Tab (cont.)

Setting	Media Type	Description
Display J# Option Dialog	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify whether the installation should display a dialog that lets end users indicate if they want to install J# on the target system. ***Note ▪*** *If the .NET Framework 1.1 is also included in the installation, the dialog states that .NET Framework 1.1 will also be installed.*
Install J# if Installation Runs Silently	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify whether you want to install J# on the target system if the J# Option dialog cannot be shown (for example, if the installation is run silently).

Internet Tab

The Internet tab is where you specify Web-related information for the WebDeployment type of release. Note that this tab is disabled for other release types.

Table 9-69 ▪ Settings on the Internet Tab

Setting	Description
Generate One-Click Install	Specify whether to create a One-Click Install, which is an installation program whose initial user interface is an HTML page. When an end user visits your Web page and clicks an Install button on it, the installation files are downloaded to the target system and then launched. If you select Yes for this setting, you must specify file names for the One-Click HTML Base Name and One-Click .cab Base Name settings.
One-Click HTML Base Name	Specify the base file name of the HTML file that InstallShield should generate at build time. The .htm file name extension is appended to the base name that you specify, and the generated file is created in the Disk1 folder of the current release location. This setting is used only if the value of the Generate One-Click Install setting is Yes.
One-Click .cab Base Name	Specify the base file name for the cabinet file (.cab) that is generated by the build process for a One-Click Install installation. The .cab file name extension is appended to the name that you specify here. The generated files are created in the Disk1 folder for the current release. The name that you specify for this setting is displayed at run time in the digital certificate when you digitally sign your installation. This setting is enabled only if the value of the Generate One-Click Install setting is Yes.

Events Tab

The Events tab lets you configure settings for distributing releases to a folder or FTP site automatically at build time or at any time on demand.

***Edition** ▪ The Events tab in the Premier edition of InstallShield has additional settings that let you specify commands that you want to be run at various stages of the build process. For example, you can specify a command that you want to be run after InstallShield has built the .msi package and the .cab files (if your product's data files are to be stored in .cab files). Note that this event occurs before the .msi package has been digitally signed and streamed into the* `Setup.exe` *file.*

Table 9-70 ▪ Settings on the Events Tab

Setting	Media Type	Description
Copy to Folder	Custom, CD-ROM, DVD, SingleImage, WebDeployment	If you want to be able to automatically distribute your release to a folder, specify the location. Existing folders with the same names as copied folders are overwritten, but no folders are deleted. Specify the folder path by entering the path in this setting, or click the ellipsis button (...) to browse to the location. ***Tip** ▪ InstallShield copies all of the relevant files for your release to the specified folder whenever you right-click the release in the Releases view and then click Distribute.* *If you want the build engine to copy your release to the specified folder after every build, select Yes for the Distribute After Build setting.* ***Note** ▪ If you specify a folder for this setting and you also specify an FTP location on the Events tab, InstallShield copies the release to only the FTP location.*

Table 9-70 ▪ Settings on the Events Tab (cont.)

Setting	Media Type	Description
FTP Location	Custom, CD-ROM, DVD, SingleImage, WebDeployment	If you want to be able to automatically distribute your release to an FTP server, specify the FTP URL for the location. Each release in your project can have a different FTP location. ***Note ▪*** *If you need to distribute your release to a path outside the FTP default folder, use a double slash (//). For example, to distribute your release to a root-level folder called* **myproduct***, where the URL of the FTP server is* **ftp://ftp.mydomain.com***, enter* `ftp://ftp.mydomain.com//myproduct` *for the FTP location.* ***Tip ▪*** *InstallShield copies all of the relevant files for your release to the specified FTP location whenever you right-click the release in the Releases view and then click Distribute.* *If you want the build engine to copy your release to the specified location after every build, select Yes for the Distribute After Build setting.*
FTP Site User Name	Custom, CD-ROM, DVD, SingleImage, WebDeployment	If a user name is required to upload to the FTP location that you are specifying, enter the user name. ***Note ▪*** *If you enter a user name for one of the releases in your project, that same user name is used for other releases in the same project and in other Express projects.*
FTP Site Password	Custom, CD-ROM, DVD, SingleImage, WebDeployment	If a password is required to upload to the FTP location that you are specifying, enter the password. ***Note ▪*** *If you enter a password for one of the releases in your project, that same password is used for other releases in the same project and in other Express projects.*
Distribute After Build	Custom, CD-ROM, DVD, SingleImage, WebDeployment	Specify whether you want the build engine to automatically distribute your release after each build to the location that you specified. ***Tip ▪*** *To quickly distribute your release on demand, instead of after each build, right-click the release and click Distribute. Note that the Distribute command is available only if the release has been built at least once.*

QuickPatch Projects

A QuickPatch project is a specific type of Windows Installer–based project recommended for installation authors who want to ship small, single upgrades to their end users. Each QuickPatch project has its own views.

Define Patch Settings View

The Define Patch Settings view is available when you open or create a QuickPatch project. Once you open or create a QuickPatch project, you can configure some of the basic settings in the Define Patch Settings view. Use this view to configure your QuickPatch.

The following table describes the views under the Define Patch Settings view:

Table 9-71 ▪ Views Under the Define Patch Settings View

View	Description
General Information	Enter information about your product and the original installation. Specify which releases should be patched by the current project, and select which custom actions should be included.
Files	Add files to your patch, and specify file-patching options.
Registry	Explore which Registry values have been added, changed, or removed.

General Information View

The General Information view contains basic information about your QuickPatch project. You can also view and configure product properties, build settings, patch history, and custom actions in this view.

Product Properties

When you click Product Properties in the General Information view, InstallShield displays the following:

Table 9-72 ▪ Settings for Product Properties

Setting	Description
Original Setup Path	This area shows the product name and the path to the original installation image. The path box is read-only.
Product Version	The Original setup version box shows the product version of the original installation. This box is read-only. In the QuickPatch Version box, you can change the product version for your QuickPatch project. Entry in this box is optional.

Table 9-72 ▪ Settings for Product Properties (cont.)

Setting	Description
FlexNet Connect Integration	If FlexNet Connect was enabled in your original installation and if you designate custom version numbers that FlexNet Connect should use to identify your product (as opposed to using the standard version scheme that Windows Installer uses), type the new custom version number for your QuickPatch in this box.

Build Settings

When you click Build Settings in the General Information view, InstallShield displays the following tabs:

- Common tab
- Identification tab
- Digital Signature tab
- Advanced tab

Common Tab

When you click Build Settings in the General Information view, InstallShield displays several tabs. The Common tab exposes frequently used build settings for a QuickPatch:

Build Location

Specify where you want your patch file built or browse for an existing folder.

Launcher Settings

In this area, you can configure the following launcher settings:

Table 9-73 ▪ Configurable Launcher Settings

Setting	Description
Create Update.exe	If you want to create an `Update.exe` update launcher for the current QuickPatch package, select this check box. To learn when an `Update.exe` update launcher is required, see Patching Considerations.
Include Windows Installer 3.1 engine	Select this check box to include the Windows Installer 3.1 engine with your patch package.
Include .NET Framework	Select this check box to include the .NET Framework with your patch package.

Patch Uninstallation

Select the **Allow Patch to be Uninstalled (Requires Windows Installer 3.0)** check box if you would like the QuickPatch to be uninstallable without having to uninstall and reinstall the entire application and other QuickPatch packages. Note that uninstallation of QuickPatch packages works only under certain conditions. For example, versions of Windows Installer earlier than version 3.0 cannot remove just the QuickPatch from an application. For more information, see Removing Patches in the Windows Installer help.

Identification Tab

When you click Build Settings in the General Information view of a QuickPatch project, InstallShield displays several tabs. The Identification tab exposes settings for display strings. The display strings are used to populate information about the patch in Add or Remove Programs. It is also used by the Windows Installer 3.0 (and later) APIs that interrogate and catalog patches applied to a target machine.

Note ▪ *Patch metadata is stored directly in the patch (.msp) file and not in any of the .msi packages. The .msp file contents are not localizable. Therefore, string table entries are not available for any of the metadata settings below.*

Table 9-74 ▪ Patch Uninstallation Settings

Setting	Description
Display name	Specify the name of your patch.
Support URL	Specify the uniform resource locator (URL) that you would like your customers to visit for technical support.
Description	Specify a brief description of your patch.
Manufacturer name	Specify the name of the application's manufacturer.
Target product name	Specify the name of the application or the target application suite.
Classification	Specify the category of upgrade. Examples include Critical Update, Hotfix, and Service Pack.

Digital Signature Tab

When you click Build Settings in the General Information view of a QuickPatch project, InstallShield displays several tabs. The Digital Signature tab is where you specify settings if you want to digitally sign your patch.

Table 9-75 ▪ Settings in the Digital Signature Tab

Setting	Description
Sign the Patch Package	To digitally sign your QuickPatch package, select this check box.
Sign Update Launcher	To digitally sign the `Update.exe` file, select this check box.

Table 9-75 ▪ Settings in the Digital Signature Tab (cont.)

Setting	Description
Use 64-Bit Signing	To digitally sign your patch package by using only the 64-bit signing framework, select this check box.
Signing Type	Specify the method to digitally sign the patch package and `Update.exe` file. Specify the argument for a sign tool's configuration: • **Standard**—Select this option to use the default InstallShield Express Edition sign tool to digitally sign the patch package and `Update.exe` file. • **Custom**—Select this option to use a customized sign tool to digitally sign the patch package and `Update.exe` file. Selecting this option enables the **Path** and **Arguments** settings. ***Note ▪*** *By default, this setting is set to* ***Standard****.*
Certificate URL	Type a fully qualified URL—for example, **http://www.mydomain.com**. This URL is used in your digital signature to link to a site that you would like end users to visit to learn more about your product, organization, or company.

Table 9-75 ▪ Settings in the Digital Signature Tab (cont.)

Setting	Description
Digital Certificate Information	To specify the digital certificate that you want to use to sign your release, click the Browse button next to this setting. The Certificate Selection dialog box opens, enabling you to specify either the location of the .pfx file, or certificate file (EV exported .cer file), or information about the certificate store that contains the certificate. InstallShield provides an option to encrypt and store an EV token password in the project file using the public key certificate (.cer) file. The .cer file is generally created by exporting a public key from the EV Authentication Client tools associated with a USB eToken provider (for example, SafeNet Authentication Client). InstallShield displays additional options to configure the Extended Validation (EV) certificate properties if the .cer file is specified. For more details, see ● Configuring Extended Validation (EV) Certificate information in InstallShield ● Certificate Selection Dialog Box After specifying the .pfx file or choosing the certificate from test store, the below will be displayed: ● **Certificate Thumbprint**—This read-only setting displays the certificate thumbprint. ● **Issued By**—This read-only setting displays the certificate issuer information. ● **Expiration Date**—This read-only setting displays the certificate's expiration date. After specifying the certificate file (EV exported .cer file), the below settings will be displayed: ● **Cryptographic Provider**—This setting allows you to specify the cryptographic service provider (CSP). ● **Container Name**—This setting allows you to specify the private key container name associated to the cryptographic service provider (CSP). ● **Token Password**—This setting allows you to specify the EV token password which is encrypted and stored in a project file. ***Note ▪*** *Note the following informations while utilizing the settings that appear after specifying the certificate file (EV exported .cer file):* ● *Both the Cryptographic Service Provider (CSP) name and Container name can be obtained from the Private Key Certificate properties of a user certificate in the EV Vendor Authentication Client tool.* ● *An EV certificate vendor determines an EV token password's expiration period and number of invalid password attempts before it is locked. Therefore, selecting this setting requires changing your password in specific intervals.* ● *If an EV token password is locked, unlocking/resetting the token password requires an administrator password.*

Table 9-75 ▪ Settings in the Digital Signature Tab (cont.)

Setting	Description
Password	If the .pfx file that you are using has a password, enter it. InstallShield encrypts the password and stores it in your project file (.ise). At build time, InstallShield uses the password to sign files with a .pfx file. If your certificate is protected by a password but you do not enter it in this setting, signing with a .pfx file fails. Note that if you configure your project to use a certificate that was imported with password protection into a store, Windows prompts for the password at build time when InstallShield is attempting to sign your project's files. The strong key protection that Windows uses does not permit InstallShield to provide the password to the cryptographic provider.
Path	Specify the sign tool's location to digitally sign the patch package and `Update.exe` file using that sign tool. To specify sign tool's location, click the ellipsis button (...) in this setting.
Arguments	Specify the command-line argument for a sign tool's configuration to digitally sign the patch package and `Update.exe` file. For example, command-line argument below can be used if the Microsoft built-in signing tool is configured as a custom option to sign the binaries: `sign /fd SHA256 /f "<ProgramFilesFolder>\testCA.pfx" /t http://timestamp.digicert.com /p MyPassword [filename]` The `[filename]` variable is a place holder for full file path to be signed. It resolves to full path of the binary file to be signed during build time. By default, a file path will be added at the end of an argument and passed to a custom sign tool. Instead of using a hard-coded path, you can use the path variables or environment variables that are defined in your project.
Signature Description	Specify the signature description that you want to use for the patch package and `Update.exe` file, if applicable. The description that you specify is displayed on the User Account Control (UAC) box to the right of the "Program Name:" label. The UAC dialog box opens when an end user launches the signed file and elevated privileges are required. If you leave this setting blank, InstallShield uses the name of the file without its extension as the description to the right of the "Program Name:" label on the UAC dialog box.

Advanced Tab

When you click Build Settings in the General Information view, InstallShield displays several tabs. The Advanced tab exposes a comprehensive set of build settings that you can configure for a QuickPatch.

Build Location

In the Build Location area, you can configure the following settings.

Table 9-76 ▪ Build Location Settings

Property	Description
Build Location	Specify where you want your patch file built or browse for an existing folder.
Create Update.exe	Specify whether you want to create an `Update.exe` update launcher for the current QuickPatch package. To learn when an `Update.exe` update launcher is required, see Patching Considerations.
List of Patch GUIDs to replace	To replace one or more earlier installed patches with the current QuickPatch, set this property to the patch GUIDs of those patches, and separate each with a comma. For example: **`{C86838C9-DEDC-4451-B96F-94AFB9460F15},{C8633E5B-AC44-45d8-B487-C68B3B1F60D6}`** Setting this property is typically *not* required, even if you have several QuickPatch projects in the History. However, if your QuickPatch project does not overwrite files added in an earlier QuickPatch project, it may be necessary to set this property. If you do not know the GUID of a patch that you want to replace, click this property and then click the ellipsis button (...). Select the patch (.msp or .exe file), and InstallShield will add the corresponding GUID to this property.
Create new "UpdatedImage" folder	This setting enables you to build your QuickPatch package with an existing UpdatedImage folder. To use this option, you must have built this package at least once (so that an UpdatedImage folder exists). Setting this option to No builds the package from the existing UpdatedImage folder. This lets you tweak the .msi package in the UpdatedImage folder and use that .msi data in a QuickPatch project. Setting this option to Yes (the default setting) regenerates the .msi file in the UpdatedImage folder every time that the package is built.
Generate MsiPatchOldAssembly tables	Specify whether to automatically generate entries for the `MsiPatchOldAssemblyFile` and `MsiPatchOldAssemblyName` tables, which allow a patch package that is running under Windows Installer 3.0 and later to patch an assembly in the global assembly cache (GAC) without making a run-time request for the original installation source. For more information, see Patching Assemblies in the Global Assembly Cache.

Table 9-76 ▪ Build Location Settings (cont.)

Property	Description
Create patch sequencing entry	Specify whether you want to use patch sequencing for your QuickPatch. A patch sequence accounts for obsolete patches, superseded patches, and patches that have already been applied to the product. The sequence specifies the order that Windows Installer version 3.0 and later should apply patches to an installed product, regardless of the order in which they are provided to the target machine. For versions of Windows Installer earlier than version 3.0, the patch sequence is ignored, and any patches are applied to the product in the order that they are provided to the target machine.
Streamline QuickPatch	Specify whether you want InstallShield to streamline the creation of your QuickPatch package to build as simple a package as possible. The default value is Yes. The goal of QuickPatch streamlining is to generate a QuickPatch package that has fewer new subfeatures and custom actions than a non-streamlined QuickPatch package. For example, if your QuickPatch project includes a new file or registry entry and InstallShield does not use QuickPatch streamlining, InstallShield creates a new subfeature for that file or registry entry. InstallShield also adds one or more prebuilt InstallShield custom actions to work around certain Windows Installer patch requirements. However, if InstallShield does use QuickPatch streamlining, the file or registry entry is added to an existing feature, and no special prebuilt InstallShield custom actions are required. ***Note ▪*** *InstallShield cannot streamline the creation of a QuickPatch package in the following scenarios:* ● *The QuickPatch package removes an installed file.* ● *The QuickPatch package removes or renames a registry key.* ● *The QuickPatch package targets a non-streamlined QuickPatch image. That is, you cannot use QuickPatch streamlining if you select the check box in the History area of the General Information view for a QuickPatch that did not use QuickPatch streamlining. If you try to build a streamlined QuickPatch that targets one or more non-streamlined QuickPatch images, InstallShield displays a build warning, and it does not use streamlining.*
Password Protect Launcher	To password-protect your QuickPatch package, select Yes, and then type a password for the Launcher Password setting. When you password-protect your QuickPatch package, any end user who wants to apply your QuickPatch must enter a case-sensitive password to launch your update. This setting is applicable only to QuickPatch packages that use an `Update.exe` file.

Table 9-76 ▪ Build Location Settings (cont.)

Property	Description
Launcher Password	Type a password to protect your application. You must select Yes for the Password Protect Launcher setting to activate password protection. When you password-protect your patch, any end user who wants to apply your patch must enter a case-sensitive password to launch your update. This setting is applicable only to patches that use an `Update.exe` file.

Windows Installer Engine

In this area, you can configure the following settings for the Windows Installer engine:

Table 9-77 ▪ Windows Installer Engine Settings

Property	Description
Include Windows Installer 3.1 engine	Specify whether the Windows Installer engine should be included with your patch package.
Engine Location	Select one of the following options: ● **Download Engine From The Web**—If the Windows Installer engine needs to be installed, it is downloaded at run time. ● **Extract Engine From Update.exe**—The build streams the Windows Installer engine into the `Update.exe` file, giving you a single file to distribute to your customers. At run time, the engine is extracted from the Update.exe file and installed if required. ● **Copy From Source Media**—The build copies the Windows Installer engine into the same directory as the Update.exe file.
Windows Installer 3.1 engine URL	Specify the URL for the location of the engine. This is the location that the `Update.exe` file uses at run time to download the engine. The default URL location is a live site maintained by Revenera for your convenience.

Microsoft .NET Framework

In this area, you can configure the following settings for the Microsoft .NET Framework:

Table 9-78 ▪ Microsoft .NET Framework Settings

Property	Description
Include In Build	Specify whether to include the Microsoft .NET Framework in your patch.

Table 9-78 ▪ Microsoft .NET Framework Settings (cont.)

Property	Description
Engine Location	Select one of the following options: • **Download Engine From The Web**—If the .NET Framework needs to be installed, it is downloaded at run time. • **Extract Engine From Update.exe**—The build streams the .NET Framework into the `Update.exe` file, giving you a single file to distribute to your customers. At run time, the engine is extracted from the Update.exe file and installed if required. • **Copy From Source Media**—The build copies the .NET Framework into the same directory as the `Update.exe` file.
Engine URL	Specify the URL for the location of the .NET Framework. This is the location that the `Update.exe` file uses at run time to download the .NET Framework. The default URL location is a live site maintained by Revenera for your convenience.

Update Launcher Settings

In this area, you can configure the following settings for the `Update.exe` launcher:

Table 9-79 ▪ Update Launcher Settings

Property	Description
Company Name	If you want to override the default company name for `Update.exe` with your company name, enter your company name. The company name is displayed on the Properties dialog box for the update launcher; this Properties dialog box opens when end users right-click the `Update.exe` file and then click Properties. To learn more, see Customizing File Properties for the Update Launcher.
Product Name	If you want to override the default product name for `Update.exe` with your product name, enter your product name. The product name is displayed on the Properties dialog box for the update launcher; this Properties dialog box opens when end users right-click the `Update.exe` file and then click Properties. To learn more, see Customizing File Properties for the Update Launcher.
Product Version	If you want to override the default product version for `Update.exe` with your product version, enter your product version. The product version is displayed on the Properties dialog box for the update launcher; this Properties dialog box opens when end users right-click the `Update.exe` file and then click Properties. To learn more, see Customizing File Properties for the Update Launcher.

Table 9-79 ▪ Update Launcher Settings (cont.)

Property	Description
Description	If you want to override the default description for `Update.exe` with your own description, enter the appropriate description. The description is displayed on the Properties dialog box for the update launcher; this Properties dialog box opens when end users right-click the `Update.exe` file and then click Properties. To learn more, see Customizing File Properties for the Update Launcher.
Copyright	If you want to override the default copyright notice for `Update.exe` with your product's copyright notice, enter your product's copyright notice. The copyright notice is displayed on the Properties dialog box for the update launcher; this Properties dialog box opens when end users right-click the `Update.exe` file and then click Properties. To learn more, see Customizing File Properties for the Update Launcher.
Required Execution Level	Use the Required Execution Level setting to specify the minimum level required by your installation's `Update.exe` file for running the installation (the setup launcher, any InstallShield prerequisites, and the .msi file) on Windows Vista and later platforms. The available options are: • **Administrator**—`Update.exe` requires administrative privileges to run. Administrators must authorize it; non-administrators must authenticate as an administrator. • **Highest Available**—`Update.exe` prefers administrative privileges. Administrators must authorize it; non-administrators run it without administrative privileges. • **Invoker**—`Update.exe` does not require administrative privileges, and all users can run it without administrative privileges. `Update.exe` does not display any UAC messages prompting for credentials or for consent. • **Use Previous Setup Manifest**—The `Update.exe` manifest uses the same required execution level that was specified for the previous setup. This is the default option. If the Create Update.exe check box is selected, InstallShield embeds an application manifest in the `Update.exe` launcher. This manifest specifies the selected execution level. Operating systems earlier than Windows Vista ignore the required execution level. If the Create Update.exe check box is cleared, InstallShield does not embed the Windows application manifest in an `Update.exe` launcher. For more information, see Minimizing the Number of User Account Control Prompts During Installation.

History

The History item in the General Information explorer within the General Information view presents a synopsis of your QuickPatch project. It lists all of the associated releases, enabling you to specify which ones should be patched by the current QuickPatch project.

Caution ▪ *If you open and modify any of the releases listed in the History, your latest project will not work since intermediate data shared across the releases in the History may have been lost or altered.*

Note ▪ *When you specify which releases from the History to patch, InstallShield looks for specific versions associated with each release to identify it.*

Custom Action

The Custom Action item in the General Information explorer within the General Information view lists the custom actions that are defined in the original installation project for which you are creating a patch. It enables you to specify which actions should be executed by the current QuickPatch project.

Files View

The Files view enables you to manage the files in your QuickPatch and also see version numbers, languages, and other information about the files in your original installation. All of the files in the Files To Patch and Original Setup Files explorers are listed by key file.

When you click a file in the Original Setup Files explorer, the file name, the destination, the version number, and other pertinent information are displayed in the right pane. You can drag and drop files from this explorer to the Files To Patch explorer, which shows all of the files that will be added, changed, or removed when your QuickPatch is applied to the original installation.

New File Settings

When you click a new file in the Files To Patch explorer within the Files view, you can configure the following settings.

File Settings

Table 9-80 ▪ Settings in the File Settings Area

Setting	Description
Specify the file that you want to add to the patch setup	If you are modifying a file in the original installation, specify the latest version of the file that you want to be installed on the target system when the patch is applied.
Self-Registering	If the file that you are adding is a self-registering DLL or .ocx file, select this check box. ***Note ▪*** *This option does not self-register .exe or .tlb files.*
Extract COM Information	If the file is a COM server and you want InstallShield to extract COM information from the file when the patch is built, select this check box.

Integration Settings

Select the file destination and the features that you want to associate with this file.

Note ▪ *This install state binding requires feature association. When you add new data to your QuickPatch project, you must associate it with a feature. New data is installed only if the corresponding feature is installed.*

Modified/Deleted File Settings

When you click a file that you have added from your original installation to the Files To Patch explorer within the Files view, you can configure the following settings:

Updated File

Table 9-81 ▪ Settings in the Updated File Area

Setting	Description
Specify the latest version of your file	If you are modifying a file in the original installation, specify the latest version of the file that you want to be installed on the target system when the patch is applied.
Include Whole File	If you want to include the selected file in your patch as a whole file, rather than only the byte-level file differences of the file, you may want to consider selecting this check box. If you do so, InstallShield configures your patch so that Windows Installer ignores any actual version number of the selected file, and instead considers it be version 1.0.0.0 when it is determining whether to update the target system's file with the version in your QuickPatch package, or to leave the file as is. Selecting this check box for an unversioned file may be helpful to ensure that the file on the target system is always overwritten with the newer equivalent unversioned file in your QuickPatch package. According to Windows Installer file overwrite rules, a file of any version is maintained over an unversioned file. Therefore, if you select this check box, Windows Installer updates the unversioned file with the one in the QuickPatch package because it considers the file in the QuickPatch to be a versioned file. ***Important ▪*** *According to Windows Installer file overwrite rules, the file with the highest version is maintained, even if the file already on the target system has a higher version than the one being installed. Therefore, if the file on the target system is a versioned file, you may want to avoid selecting this check box, since in some scenarios it could lead to the file on the target system not being updated. For example, if the file on the target system is version 1.1.0.0, the file in your QuickPatch project is version 2.0.0.0, and you select this check box, the file on the target system is not updated at run time. This occurs because even though the actual version number of the file in the QuickPatch package is newer than the one on the target system, Windows Installer considers the file in the QuickPatch to be 1.0.0.0, which would make it older than the file on the target system.*

Table 9-81 ▪ Settings in the Updated File Area (cont.)

Setting	Description
Extract COM Information	If the file is a COM server and you want InstallShield to extract COM information from the file when the patch is built, select this check box. This check box is selected by default for patch files that contain COM information in the `Class` or `TypeLib` tables in the base .msi package. ***Note ▪*** *When you specify an existing file that you want to patch, InstallShield automatically detects if the file is self-registering. If the original file was self-registering, the file in the patch is also set to self-register.*

Original File Information

The Original File Information area provides information about the original file in the original installation.

Check this box to have the patch delete this file from the setup

Select this check box to remove the file from the target machine when the patch is applied.

Registry View

The Registry view in a QuickPatch project provides a visual representation of what currently exists on a source machine and what you will patch on a destination/target system once your patch project is built and applied to the target system. The Destination computer's Registry view pane and the Destination computer's registry data pane are prepopulated with registry entries in the original installation.

Each item in the Destination computer's registry data pane appears next to an icon specific to the value type and the data's current condition. Icons for data values that have been modified have a turquoise pencil. Icons for new data values have a red star in the upper-right corner. Icons for registry values that will be deleted from the target system have a red X. See the following sample icons and descriptions below.

Table 9-82 ▪ Icons in the Registry View of a QuickPatch Project

Icon	Description
	This icon represents a new DWORD value that will be added to the target system when the QuickPatch is applied.
	This icon represents a modified DWORD value that will be updated on the target system when the QuickPatch is applied.
	This icon represents an existing DWORD value on the target system from an earlier installation.
	This icon represents a DWORD value that exists on the target system but will be deleted when the QuickPatch project is applied.

Errors and Warnings

The following topics provide information about errors and warnings that might occur when you are working with your installation.

Table 9-1 ▪ List of Error and Warning Topics

Topic	Description
Build Errors and Warnings	Lists errors and warnings that might occur during the build process.
Upgrade Errors and Warnings	Lists potential errors and warnings that may occur if you upgrade a project created with an earlier version to the current version of InstallShield.
Windows Installer Run-time Errors	Lists Windows Installer errors that might occur during run time. The errors are related to the built-in InstallShield custom actions that are added automatically to InstallShield projects to support some types of functionality.
Setup.exe Run-Time Errors and Warnings	Lists the errors that might occur when `Setup.exe` runs.
Visual Studio Project Import Errors and Warnings	Lists the errors and warnings that might occur when you do either of the following: • Convert a Visual Studio setup project (.vdproj) to an InstallShield project (.ise). • Import a Visual Studio setup or merge module project (.vdproj) into an InstallShield project (.ise).

To find information about Windows Installer run-time errors, see Windows Installer Error Messages in the Windows Installer Help Library.

Build Errors and Warnings

The table below provides troubleshooting tips for each build error and warning.

Table 9-2 ▪ Build Errors and Warnings

Error	Description	Troubleshooting Tips
-32000	Build canceled by the user.	This error occurs only when the build is terminated during the build process.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-7347	InstallShield does not support signing with .spc and .pvk files. Replace the digital certificate information in your project with a .pfx certificate, or with a reference to a certificate in a certificate store.	This error occurs if you added .spc and .pvk files to your project in an earlier version of InstallShield that included support for digitally signing with those files, and then you open your project in InstallShield 2026 Express Edition and attempt to build a release or a patch. To resolve this error, remove the .spc reference from the release or QuickPatch project. You can replace it with a .pfx certificate, or with a reference to a certificate in a certificate store. To learn more, see: ● Digital Signing and Security ● Digitally Signing a Release and Its Files at Build Time ● Signing a QuickPatch Package
-7346	The files in this release are being signed with a SHA-1 certificate. Windows will not trust files that were signed with SHA-1 certificates if they were signed after January 1, 2016.	This warning occurs if your project is configured to use a SHA-1 certificate to digitally sign your release at build time. SHA-256 is favored over SHA-1, which is being deprecated because of the potential for security vulnerabilities. Thus, it is recommended that you replace any SHA-1 certificates in your InstallShield projects with SHA-256 certificates. In InstallShield, to replace a SHA-1 certificate with a SHA-256 certificate for signing your releases, use the Signing tab in the Releases view to replace the reference to the current certificate with one for a SHA-256 certificate. To learn more, see Digital Signing and Security.
-7256	To digitally sign a software identification tag file, .NET Framework 3.5 must be installed.	If your project is configured to include a software identification tag and if the release that you are building is configured to use a .pfx file to digitally sign your release, InstallShield attempts to digitally sign the tag that it creates at build time. If your build machine does not have .NET Framework 3.5, this build warning is displayed. To resolve this build warning, install .NET Framework 3.5 on your build machine.
-7255	Only .pfx files can be used to digitally sign a software identification tag file.	If your project is configured to include a software identification tag and if the release that you are building is configured to use a certificate in a certificate store to digitally sign your release, InstallShield generates this build warning. To resolve this build warning, switch from a certificate in a certificate store to a .pfx file. If you do not need to have a digitally signed tag, you can ignore this warning.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-7236	InstallShield could not create the software identification tag because the value for the %1 setting in the General Information view is invalid. Ensure that it does not contain any of the following characters: \ / : * ? " < > |	InstallShield uses the values that you enter for the Unique ID and Tag Creator ID settings as part of the name of the tag file (*TagCreatorID_UniqueID*.swidtag). Therefore, the ID that you enter must not contain any characters that are invalid for file names. To resolve this issue, enter a valid value in the setting that is mentioned in the warning message. For more information, see Including a Software Identification Tag for Your Product.
-7235	InstallShield could not create the software identification tag because the %1 setting in the General Information view is empty.	Creation of a software identification tag requires that several settings in the General Information view have values. To resolve this warning, enter the appropriate value in the setting that is mentioned in the warning message, or select No for the Use Software Identification Tag setting. For more information, see Including a Software Identification Tag for Your Product.
-7234	InstallShield could not create the software identification tag because the template file `Swidtag.xml` could not be opened.	InstallShield uses the `Swidtag.xml` file, which is installed to one of the following locations, to build tag files: *InstallShield Program Files folder*\Support\0409 *InstallShield Program Files folder*\Support\0411 To resolve this issue, ensure that the `Swidtag.xml` file is present and not locked. For more information, see Including a Software Identification Tag for Your Product.
-7222	No supported languages included in the media. You must select at least one supported language for the media.	This error occurs only when no supported language exists in the media.
-7211	Building a compressed Network Image Setup.exe. All other build types are not allowed in the evaluation mode of InstallShield.	If you are using InstallShield in evaluation mode (that is, you have not activated it), you can build compressed `Setup.exe` files, but no other release types. If you are creating a Windows Installer–based release, the .msi database is always embedded in the `Setup.exe` file. If you try to build an uncompressed release in evaluation mode, the uncompressed option is ignored, and InstallShield displays this build warning. Once you have activated InstallShield, the evaluation-mode limitations are removed.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-7209	The current Windows Installer package name includes Unicode characters that may cause a run-time error if a compressed Network Image Setup.exe is built.	This build warning occurs if the following conditions are true: ● For the name of the .msi package, you include Unicode characters that are not supported by your build machine's ANSI code page. ● You include a setup launcher (`Setup.exe` file) with your release. This warning occurs to inform you that when the Windows Installer package is extracted from `Setup.exe` to a temporary location on the target system, run-time error 1152 may occur. To resolve this warning, change the name of the .msi package so that it does not contain Unicode characters that are not supported by your machine's ANSI code page. To do so, you can use the MSI Package File Name setting, which is available when you click the Express node in the Releases view.
-7208	The current location for this package includes Unicode characters that may cause a run-time error when Windows Installer tries to extract its .cab files.	This warning occurs if the following conditions are true: ● The path for the release that InstallShield is building contains Unicode characters that are not supported by your build machine's ANSI code page. ● You configure the release so that your product's program files are compressed into .cab files. You can ignore this warning if end users will launch your release from a location whose path does not contain Unicode characters that are not supported by your build machine's ANSI code page. If end users try to launch your installation from a path that contains Unicode characters that are not supported by the target system's ANSI code page, run-time error 1311 occurs when Windows Installer tries to extract the .cab files.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-7185	The %1 translation for string identifier %2 includes characters that are not available on code page %3.	This error occurs if a string value that you entered for the specified language contains characters that are not available in that language's code page. To resolve this error, use the Text and Messages view to edit the value of the string so that it uses characters from the appropriate code page. ***Edition ▪*** *The Premier and Professional editions of InstallShield enable you to build releases that use UTF-8 encoding for the .msi database. The UTF-8 encoding supports characters from all languages simultaneously, enabling you to mix and match, for example, Japanese and German, or Russian and Polish, both in text shown to end users and in file names and registry keys. These mixed languages work correctly regardless of the current language of the target system.*
-7184	The %1 column of the %2 table includes characters that are not available on code page %3: "%4"	This error occurs if a part of your project contains characters that are not available in the code page for one or more target languages. To resolve this error, change the data that is described in the error message so that it uses characters from the appropriate code page. For example, if error message mentions the `Shortcut` table, consider changing the string in the Shortcuts/Folders view. ***Edition ▪*** *The Premier and Professional editions of InstallShield enable you to build releases that use UTF-8 encoding for the .msi database. The UTF-8 encoding supports characters from all languages simultaneously, enabling you to mix and match, for example, Japanese and German, or Russian and Polish, both in text shown to end users and in file names and registry keys. These mixed languages work correctly regardless of the current language of the target system.*

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-7128	Error embedding setup.exe.manifest.	This error occurs if there is a problem embedding the application manifest in the `Setup.exe` launcher. The application manifest specifies the minimum privilege level required by your installation's `Setup.exe` file for running the installation (the setup launcher, any InstallShield prerequisites, and the .msi file) on Windows Vista and later platforms. This error may occur if the template manifest file is missing from the InstallShield `Support` folder. It also may occur if the `Setup.exe` template is missing from the InstallShield `Redist\Language Independent\i386` folder. To resolve this error, try running a repair of InstallShield.
-7124	Could not invoke MSBuild. Make sure that the .NET Framework %1 is installed at '%2' on the system. If you are running InstallShield from within Visual Studio, you must use Visual Studio 2005 or later.	If you use MSBuild to build Visual Studio solutions with InstallShield projects, you must have .NET Framework 3.5 or later on your machine. In addition, you must be using Visual Studio 2005 or later.
-7113	Language support for %1 is not included in this edition.	This error appears if you try to enable too many languages in a project, building with more than two in a non-Premier edition. The build is successful (unless stop at first error is enabled), but the extra languages are not built. The first two qualified languages are built. %1 is replaced with the name of the language that is not included in this release. To resolve this error, use the Premier edition of InstallShield to build your release.
-7084	The VBScript Custom Action %1 does not point to a valid VBS file.	This warning message is displayed if you add a VBScript custom action to your installation but the file specified for the custom action is not a VBScript file. To resolve this error, select the appropriate type of file for the specified custom action in the Custom Actions view.
-7076	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-7075	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-7074	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-7073	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-7072	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-7071	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-7069	Could not create XML reading object (MSXML3.dll). Please make sure this file is registered on the system.	Check the Knowledge Base for information about this error, or request technical support.
-7064	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-7063	The File %1 in Component %2 is being installed to the Windows Fonts folder, but there is no corresponding record in the Font table. The Font will not be registered properly on the target system.	To resolve this warning, open the Files view and remove the font mentioned in the message. Then re-add the font to the Fonts folder; InstallShield adds the corresponding record to the Font table automatically.
-7062	Refreshing the COM+ settings from the client machine. An error occurred while refreshing the COM+ settings from the client machine.	This build error occurs when an exception error has occurred with a COM+ application that is selected in the Component Services view. An option is provided to refresh the COM+ settings during the build process, and build error -7062 occurs when this option is selected and the refreshing process has failed. To resolve this error, you can open the Component Services view, select the COM+ application, and clear the Refresh the COM+ settings from the client machine at build check box on the Installation tab.
-7061	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-7060	Due to licensing requirements, InstallShield requires that you provide the MSI 3.0 engine redistributable from the MSI 3.0 Beta. Please copy instmsi.exe from the MSI 3.0 Beta folder to \redist\Language Independent\i386\MSI3.0\instmsiw.exe.	Check the Knowledge Base for information about this error, or request technical support.
-7059	Unable to update the latest patch version property. As a result, future patches built from this project may not be sequenced properly. Make sure the project is not marked as read-only.	Check the Knowledge Base for information about this error, or request technical support.
-7058	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-7057	You have enabled FlexNet Connect for this deployment, however, you have not provided a URL to check for updates. This is required for 'CD-ROM' deployment configurations.	Check the Knowledge Base for information about this error, or request technical support.
-7053	The 'Internet' deployment configuration is enabled, but the 'install location' field has not been authored. This will result in errors when running your deployment.	Check the Knowledge Base for information about this error, or request technical support.
-7052	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-7051	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-7050	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-7049	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-7048	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-7045	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-7044	An error occurred while copying the main application manifest to the release specific location.	Check the Knowledge Base for information about this error, or request technical support.
-7042	An error occurred while determining the source location for the primary application assembly.	Check the Knowledge Base for information about this error, or request technical support.
-7041	Initializing the default deployment manifest.	Check the Knowledge Base for information about this error, or request technical support.
-7040	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-7038	An error occurred while creating the base deployment folders in the release location.	Check the Knowledge Base for information about this error, or request technical support.
-7037	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-7036	An error occurred while loading the primary application assembly.	Check the Knowledge Base for information about this error, or request technical support.
-7035	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-7030	An error occurred while loading the releases that need to be built.	Check the Knowledge Base for information about this error, or request technical support.
-7029	The build engine is unable to load the settings for the 'CD-ROM' release type.	Check the Knowledge Base for information about this error, or request technical support.
-7028	The build engine is unable to load the settings for the 'File Share' release type.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-7027	The build engine is unable to load the settings for the 'Internet' release type.	Check the Knowledge Base for information about this error, or request technical support.
-7026	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-7025	The COM+ component server file '%1' cannot be found on your system. The component '%2' will not be installed on the target system. You will need to install the component manually after your COM+ application is installed.	%1 refers to the COM+ DLL that is missing, and %2 refers to the ProgID. To resolve this warning, add the DLL to your system in the appropriate location, or remove the COM+ component from your project.
-7024	To include J# in the build you must also include .NET 1.1.	J# requires the .NET Framework Redistributable Package version 1.1, but it is not included in the installation. If you are sure that your end users already have the .NET Framework 1.1 on their systems, you can ignore this warning message. However, if there is a chance that they will not have the .NET Framework 1.1, add it to your installation. If the .NET Framework is not on their systems, they might have trouble installing J# as part of your installation.
-7023	Internal build error.	This error occurs if a merge module is not configured correctly or it is corrupted. If you have more than one merge module in your project but you do not know which one is causing the error, you need to first identify the problematic merge module. You can do this by removing a merge module from your project and then building your release. If the error recurs, you have identified the problematic merge module. If not, remove a different merge module and then build. Once you have identified the merge module causing the error, try reinstalling it. If that does not resolve the error, contact the author to obtain the latest version.
-7012	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-7011	%1 must have a strong name to be installed to the GlobalAssemblyCache.	You can install an assembly to the Global Assembly Cache only if it has a strong name. A strong name contains the assembly's simple text name, version number, culture information (if available), a public key, and a digital signature. Using strong-named assemblies in the Global Assembly Cache enables you to have multiple versions of an assembly with the same name but with different versions of .dll files. The assemblies stored in the Global Assembly Cache can be shared by different applications on a computer. To resolve this error message, you have two options. • Change the target destination for the assembly to a location other than the Global Assembly Cache if sharing that assembly with other applications is not explicitly required. Note that it is not necessary to install assemblies into the Global Assembly Cache to make them accessible to COM interop. • Select an assembly that has a strong name for installation to the Global Assembly Cache. Note that once an assembly is created, you cannot sign it with a strong name. You can sign an assembly with a strong name only when you create it.
-6654	The file %1 is not the key file of component %2. If you call a function in a standard DLL that is installed with the product and the action is scheduled as deferred, the DLL you are calling must be the component's key file.	This error occurs if either of the following is true: • InstallShield has not set this file as the key file of a component because the file is not considered to be a PE file. • InstallShield has not set this file as the key file of a component because the file is dynamically linked. A dynamically linked file cannot be set as a component's key file. To resolve this build error, do one of the following: • Ensure that DLL is listed as a PE file on the File Extensions tab of the Options dialog box. • Exclude the file from the folder being dynamically linked, and then add the file to your project without linking it dynamically.
-6653	The feature %1 in the installation contains more than 1600 components. There is a maximum limit of 1600 components per feature.	To resolve this error, split up the feature that is identified in the error message into subfeatures, and then reassign the feature's files to these subfeatures.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6651	The setup you are building contains more than 32,767 files. Automatically switching setup package to appropriate MSI schema.	This warning appears if your installation package contains more than 32,767 files. InstallShield automatically applies the large package schema. ***Caution ▪*** *There is an issue with patching if your package uses the large package schema. When you build the patch, the following error is triggered: "Can't Generate Primary Transform."*
-6647	Cannot move file from '%1' to '%2'	Check the Knowledge Base for information about this error, or request technical support.
-6646	The '%1' property in the merge module '%2' is set to NULL. This property cannot be NULL.	Check the Knowledge Base for information about this error, or request technical support.
-6645	%1 failed to load. Error: %2 Error Description: %3	Check the Knowledge Base for information about this error, or request technical support.
-6641	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6640	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6639	The merge module '%1' requires one of the following merge modules also to be included in the setup: %2	Check the Knowledge Base for information about this error, or request technical support.
-6638	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6637	Invalid registry data for component NewComponent1 while importing *"C:\RegTest\SingleSlashl"=dword:2* (where the information in italics represents the invalid data).	The invalid data in this example is the use of a single backslash instead of two backslashes. Another example would be an entry with *DWORD* in it instead of *dword*.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6636	The file key %1 and %2 are found in the File table. Despite having different cases, the identical key names will cause an unexpected result when the files are installed on the target system. This occurs because the compressed files in the cabinet file are named using the file keys. To correct this issue, change one of the file keys to be unique in the cabinet file if you are building a compressed setup or a merge module. You can change the file key name in the Direct Editor view.	Check the Knowledge Base for information about this error, or request technical support.
-6635	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6634	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6633	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6632	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6624	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6623	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6622	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6620	Could not open the reference package '%1' for key synchronization.	Check the Knowledge Base for information about this error, or request technical support.
-6619	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6618	An error occurred while setting the key paths for dynamically created components.	Check the Knowledge Base for information about this error, or request technical support.
-6617	An Error occurred while adding the file '%1' to the synchronization maps.	Check the Knowledge Base for information about this error, or request technical support.
-6616	An error occurred while setting the key path for component '%1'.	Check the Knowledge Base for information about this error, or request technical support.
-6615	Could not update File.FileName for File '%1'.	Check the Knowledge Base for information about this error, or request technical support.
-6613	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6612	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6611	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6558	The Custom Action %1 in the InstallExecuteSequence table is run from an installed file. To run the custom action successfully, you may need to have a condition that checks if the source file is installed locally.	To successfully run a custom action that is run from an installed file, you must ensure that the file is installed locally using a conditional statement: ● If the custom action is sequenced before RemoveFiles—Run the action only if the feature was installed locally. ● If the custom action is sequenced between RemoveFiles and InstallFiles—Run the action only if the feature was installed locally. Do not run the action on an uninstallation. ● If the custom action is sequenced after InstallFiles—Run the action only if the feature will be installed locally.
-6557	Unable to find the .NET Compact Framework for the given platform/processor combination from <InstallShield>\Support\NetCF.ini file.	Ensure that the .NET Compact Framework is in the correct location.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6556	Error in including .NET Compact Framework .cab files in setup.	This error might occur for the following reasons: ● The .NET Compact Framework files cannot be found in the locations specified in the *InstallShield Program Files Folder*\Support\.NetCR.ini file. ● The CEDefault.ico file is not in the *InstallShield Program Files Folder*\Program\04xx folder. ● There is an error writing to the ISCEInstall table.
-6555	Unable to copy <InstallShield>\Support\CFApp Mgr.ini file to a temporary location.	Make sure that CFAppMgr.ini file exists at *InstallShield Program Files Folder*\Support folder. If this file does not exist, launch the InstallShield setup in repair mode. Remove any temporary files to create more disk space on the system.
-6553	Unable to find the .NET Compact Framework .cab file for the processor.	Open the *InstallShield Program Files Folder*\Support\NetCF.ini file and configure the .NET CF selection for this processor to resolve the issue.
-6551	The Name and Value columns are blank in the registry record Registry1 of Component NewComponent1. The Windows Installer will set the (Default) value of HKEY_LOCAL_MACHINE\New Key #1 to an empty string. If you would like to set the (Default) value to (Value Not Set), you need to set “Install if absent, Uninstall if present” for that key in the Registry view. If the key already exists on the Target machine, the (Default) value will not be changed.	This warning appears for every registry key you add to your project and do not manually set the default key. To set the (Default) value to (Value Not Set), you need to set “Install if absent, Uninstall if present” for that key in the Registry view.
-6550	%1 is not a valid .NET assembly to be installed to the Global Assembly Cache (where %1 is replaced with a full file path).	Only valid assemblies can be installed to the Global Assembly Cache. Ensure that the file selected for installation is a valid assembly.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6530	Error processing setup [1] for Upgrade Item [2]. This does not appear to be a valid setup.	This error occurs when an upgrade entry exists in the Upgrade Paths view, but the earlier installation that is specified does not exist in the location indicated. This can occur if the earlier installation has been moved because it was referenced in the Upgrade Paths view or if the folder that contains the earlier installation cannot be accessed because of permissions. This error can also occur if the earlier installation referenced (Setup.exe or .msi) is not a valid Windows Installer–based installation.
-6525	The custom action %1 in the InstallExecuteSequence table is run from an installed file. It must be sequenced after the %2 action. Ensure that the custom action is sequenced properly and that the base action exists in the sequence.	This error occurs when you have authored a custom action that runs from a file that is installed during your setup. The custom action must be launched during the After Setup Complete Success dialog part of the installation. Otherwise the file will not exist on the target system when Windows Installer runs the custom action.
-6499	The Shallow folder structure setting should not be used with multi-disk releases.	The Shallow folder structure setting can be set from the Releases property grid of the Releases view. If error -6499 appears, switch the Shallow folder structure option to "No" so it will create a regular build folder structure.
-6479	Internal build error.	A corrupted merge module in your project causes this error. The error message has been reported with third-party merge modules that are not shipped with InstallShield or merge modules that are shipped with the beta version of Microsoft Visual Studio .NET 2003. To resolve this error message, determine the merge module causing the issue and remove it from the project. To find the merge module, remove all merge modules from your project in the Redistributables view. Then add one merge module at a time to your project, and build the release until error -6479 occurs. This is the merge module causing the error. After you determine which merge module causes the issue, contact the vendor of the merge module for more information about the module. If the merge module is a Microsoft-specific module, download the latest service pack of Visual Studio to acquire the updated merge modules. If the merge module is Visual Studio .NET-specific, make sure that it is acquired from the latest service pack of Visual Studio .NET.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6478	FON Files must have a Font Title. Font %1 in Feature %2.	Set the Font Title property for the file. To do this, right-click the file in the Files view, select Properties, and type a title in the Font Title field. If there is no Font Title, a run-time error occurs upon installation.
-6274	Setup.exe is not found in the Disk1 location: %1. Please make sure that the previous full build of your release was completed if you are running the Build Tables Only.	This error occurs when the setup attempts to get the stamp from Setup.exe, but Setup.exe does not exist in the Disk1 folder. This happens when you cancel a full build in the middle of the process, and then run the Build Tables Only option.
-6271	File %1 not found. An error occurred building the MsiFileHash table record for this file. Verify that the file exists in the specified location.	Ensure that the identified file exists in the specified location. If the file includes the predefined path folder VSSolutionFolder, see Using the VSSolutionFolder Path Variable with Visual Studio Solutions.
-6270	The record %1 in the Icon table exceeds the limit of %2 characters. As a result, the build will be unable to persist the database.	Make sure that the record is at or under the limited number of characters.
-6269	An error occurred copying directory %1 to %2. Please ensure that the source directory's path is correct.	Check the source directory's path.
-6268	Internal build error	Check the Knowledge Base for information about this error, or request technical support.
-6267	An error occurred while extracting files from the cab file %1 to the location %2	Ensure that the location is accurate.
-6266	An error occurred while attempting to create and initialize the CAB extraction engine for %1	This error can occur when one or more files is not registered. Ensure that all files are registered.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6265	An error occurred building feature '%1'. This feature has the same name (with different case) as another feature in the project. Duplicate feature names are not allowed in Standard projects. Rename one of the features to fix this error.	Remove or rename one of the features.
-6264	A record in the %1 table is using string ID '%2' for column '%3' but this string is blank and the column is not nullable	Provide a string for use in the column.
-6260	Internal build error	Check the Knowledge Base for information about this error, or request technical support.
-6259	Internal build error	Check the Knowledge Base for information about this error, or request technical support.
-6258	An error occurred extracting digital signature information from file "%1". Make sure the digital signature information provided in the IDE is correct.	Ensure that the digital signature information is correct. You can access the digital signature information on the Signing tab in the Releases view.
-6257	The build engine has detected that PRODUCT_NAME has been defined in your project. At run time, your installation will display this value as your Product Name instead of the Product Name values defined in the IDE.	This warning message occurs when the ID PRODUCT_NAME has been defined in the string table. This string table ID overrides the Product Name values set in the General Information view. Check article Q109136, "ERRDOC: Build Error -6257, in the Knowledge Base for information on how to resolve this error.
-6255	Internal build error	Check the Knowledge Base for information about this error, or request technical support.
-6254	An error occurred building the MsiFileHash table for File %1	Check the Knowledge Base for information about this error, or request technical support.
-6253	Internal build error	Check the Knowledge Base for information about this error, or request technical support.
-6252	Internal build error	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6251	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6250	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6248	Could not find dependent file %1 of feature %2	The missing file is required by the feature's key file. Make sure the missing file is on the build system in the same directory as the feature's key file. If the dependency is not required, you can turn off .NET dependency scanning for the feature. To turn off dependency scanning, set the feature's .NET Scan at Build property to None or Properties Only.
-6247	The .NET Framework redistributable %1 is not found on the system. %2	Check the Knowledge Base for information about this error, or request technical support.
-6245	One or more of the project's features contain .NET attributes that require the .NET Framework. It is recommended that the release include the .NET Framework.	Add the .NET Framework to your project by first selecting the media type in the Releases view. When you select the media type, the build properties are displayed. In the .NET Framework Location property, select the appropriate option: • **Copy from Source Media**—Select this option to leave the .NET Framework runtime on the root of the source medium. This option is not available for Web media types or Network Image media types with all of your files compressed into Setup.exe. • **Extract from Setup.exe**—Select this option to compress the .NET Framework runtime into Setup.exe, to be extracted at run time. • **Download from the Web**—Select this option to download the .NET Framework runtime (if necessary) from a URL that you specify. If you select this option, you must set the .NET Framework URL property to the appropriate Web location. In the .NET Framework Version property, select the .NET version that you want to distribute with your installation.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6244	One or more of the project's features contain .NET attributes that require the .NET Framework. However, the .NET Framework cannot be detected. %2	Download the InstallShield .NET Update from http://www.installshield.com/products/dotnet.asp. If you have the Visual Studio release candidate, you can copy dotnetfx.exe from Windows Feature Update CD, in the dotnetframework directory, to *InstallShield Program Files Folder*\Redist\0409\i386.
-6243	InstallUtilLib.dll cannot be located on your system.	This file is required for Installer custom actions. You may need to re-run the setup to (re)install the Microsoft .NET Framework redistributable. You can download this file from http://www.installshield.com/products/dotnet.asp, or you can copy dotnetfx.exe from Windows Feature Update CD, in the dotnetframework directory, to *InstallShield Program Files Folder*\Redist\0409\i386 (if you have the Visual Studio release candidate).
-6242	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6241	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6240	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6239	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6238	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6237	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6236	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6235	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6234	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6233	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6232	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6231	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6230	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6229	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6228	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6227	.NET scan of Feature %1 failed	Check the Knowledge Base for information about this error, or request technical support.
-6226	Opening Visual Studio .NET solution	Make sure that the solution exists at the location specified. In addition, ensure that Visual Studio is installed on the build machine.
-6225	Resolving Visual Studio .NET project output "%1"	Make sure the project exists in the solution. In addition, ensure that project has the specified output.
-6224	Processing merge module %1 of feature %2	Check the Knowledge Base for information about this error, or request technical support.
-6223	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6222	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6221	Could not resolve Visual Studio .NET project output "%1" from feature %2	Make sure the project exists in the solution. In addition, ensure that project has the specified output.
-6219	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6218	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6217	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6216	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6215	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6214	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6213	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6212	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6211	Destination of Feature %1 is GlobalAssemblyCache but key file is not a .NET assembly	Check the Knowledge Base for information about this error, or request technical support.
-6208	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6207	An error occurred building Setup File %s.	Check the Knowledge Base for information about this error, or request technical support.
-6205	An error occurred building Feature %1. The Feature's destination directory %2 is not found in the directory table. Change the Feature's destination to a valid location.	Check the Knowledge Base for information about this error, or request technical support.
-6204	An error occurred importing %1 for Feature %2. Make sure the file exists in the specified location and that the file is a valid REG file.	Check the Knowledge Base for information about this error, or request technical support.
-6203	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6202	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6201	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6200	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6199	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6197	An error occurred streaming %1 into %2. Check to make sure the .msi package is not currently in use by another process.	Ensure that the .msi package is not currently being used by another process.
-6196	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6195	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6194	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6193	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6192	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6191	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6190	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6189	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6188	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6187	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6186	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6185	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6184	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6183	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6182	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6181	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6180	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6179	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6178	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6177	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6176	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6175	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6174	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6173	An error occurred renaming file %1 to %2	Make sure that you have enough free hard disk space on the drive that contains the interim folder under your product configuration (for example, \MySetups\Your Project Name-27\Product Configuration 1\Interm).
-6172	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6171	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6170	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6169	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6168	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6167	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6166	An error occurred updating the Word Count Summary Property of the Summary Information Stream.	Check the Knowledge Base for information about this error, or request technical support.
-6165	An error occurred exporting table %1.	Check the Knowledge Base for information about this error, or request technical support.
-6164	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6163	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6162	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6161	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6160	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6158	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6156	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6155	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6154	Build canceled by the user.	This error occurs only when the build is terminated during the build process.
-6153	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6152	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6151	Cannot save target database.	Check the Knowledge Base for information about this error, or request technical support.
-6150	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6149	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6148	Cannot insert record in the specified table.	Check the Knowledge Base for information about this error, or request technical support.
-6147	Cannot update the specified target field in the table.	Check the Knowledge Base for information about this error, or request technical support.
-6146	Cannot create new record in the specified table.	Check the Knowledge Base for information about this error, or request technical support.
-6145	Cannot retrieve value of the specified column in the table.	Check the Knowledge Base for information about this error, or request technical support.
-6144	Cannot open database view for table %1.	Check the Knowledge Base for information about this error, or request technical support.
-6143	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6142	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6141	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6140	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6139	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6137	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6136	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6135	The specified feature is not associated with any Setup Type.	Associate the specified feature with a setup type.
-6134	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6133	An error occurred updating properties specified in the current Product Configuration.	Check the properties specified in the current product configuration to ensure they are valid.
-6131	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6130	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6129	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6128	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6127	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6126	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6125	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6124	Unknown Exception.	Check the Knowledge Base for information about this error, or request technical support.
-6123	Unknown Exception.	Check the Knowledge Base for information about this error, or request technical support.
-6122	No features are included in the build.	Ensure your build contains at least one feature.
-6121	Error occurred copying built Merge Modules to the Modules folder.	Check the Knowledge Base for information about this error, or request technical support.
-6120	The Resource Linker failed to build the specified DLL.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6118	The Resource compiler failed to build the specified RES file required to link the DLL.	Check the Knowledge Base for information about this error, or request technical support.
-6116	Failed to export .rc file from project: %1.	Check the Knowledge Base for information about this error, or request technical support.
-6115	Could not retrieve version of IDriver.exe from ISScript.msi.	Check the Knowledge Base for information about this error, or request technical support.
-6114	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6113	An error occurred during the incremental build.	Check to ensure that a previous media was built before selecting "Build Tables Only" or "Build Tables and Refresh Files" and ensure the previous media was not deleted.
-6112	Error deleting a file streamed into Setup.exe.	Check the Knowledge Base for information about this error, or request technical support.
-6111	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6110	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6109	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6108	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6107	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6106	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6105	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6104	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6103	Could not find file %1.	Check the file to ensure it exists in the specified location. If the file includes the predefined path folder VSSolutionFolder, see Adding References to Visual Studio Solutions.
-6102	Error searching for dynamic files matching "%1".	Make sure the dynamic file flag specified is valid.
-6101	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6100	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6099	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6098	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6097	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6096	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6095	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6094	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6093	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6092	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6091	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6090	The scrollable text file:'%1' specified for the Control:'%2' does not exist.	This build error is caused by a scrollable text control that (1) is specified for the License Agreement dialog box or the Readme dialog box and (2) is not pointing to a valid .rtf file. To resolve this error message, open the Dialogs view, expand the Dialogs node, and then select the License Agreement node. Set the License File property to \Redist\0409\Eula.rtf. Then select the Readme node. Set the Readme File property to \Redist\0409\Readme.rtf. As an alternative, you can set the License File property or the Readme File property to your own file with the proper path. As long as the builder is able to find the file in the path specified in this property, the error should not occur.
-6088	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6087	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6086	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6085	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6084	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6083	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6082	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6081	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6080	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6079	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6078	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6077	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6076	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6075	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6074	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6073	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6072	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6071	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6070	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6069	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6068	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6067	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6066	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6065	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6064	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6063	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6062	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6061	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6060	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6059	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6058	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6057	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6056	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6055	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6054	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6053	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6052	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6051	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6050	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6049	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6048	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6047	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6046	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6045	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6044	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6043	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6042	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6041	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6040	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6039	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6038	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6037	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6036	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6035	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6034	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6033	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6032	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6031	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6030	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6029	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6028	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6027	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6026	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6025	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6024	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6023	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6022	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6021	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6020	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6019	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6018	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6017	The build was unable to extract COM information; make sure that you are running as Administrator.	Extracting COM information in InstallShield requires administrative privileges. For more information, see Launching InstallShield with vs. Without Administrative Privileges.
-6016	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6015	Error generating short file name for file: %1.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6014	The shortcut for a feature is invalid because it does not reference an icon resource.	Specify an icon file and icon index for this shortcut to resolve this issue.
-6013	The feature condition for the specified feature is invalid.	Use the Features view to modify the condition property of the specified feature.
-6012	Failed to set codepage for language %1.	Install the codepage for the specified language.
-6011	Failed to open string table.	Check the Knowledge Base for information about this error, or request technical support.
-6009	An error occurred creating the Web media for Internet Explorer.	Check the Knowledge Base for information about this error, or request technical support.
-6008	An error occurred creating the Web media for Internet Explorer.	Check the Knowledge Base for information about this error, or request technical support.
-6007	Error occurred copying user specified uncompressed setup files to Disk1.	Check the Knowledge Base for information about this error, or request technical support.
-6006	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-6005	An error occurred creating the Package Definition File.	Check the Knowledge Base for information about this error, or request technical support.
-6004	An error occurred streaming the digital certificate for Msi engine(s) into setup.exe.	Check the Knowledge Base for information about this error, or request technical support.
-6003	An error occurred streaming '%1' into setup.exe.	Check the Knowledge Base for information about this error, or request technical support.
-6002	Error while attempting to run the custom build setup for objects.	Check the Knowledge Base for information about this error, or request technical support.
-6001	An Error occurred while preprocessing ISObjectProperty. Make sure that ll items have a valid IncludeInBuild tag.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-6000	An Error occurred gathering information about supplemental merge modules.	Check the Knowledge Base for information about this error, or request technical support.
-5099	An error occurred creating the build report.	Check the Knowledge Base for information about this error, or request technical support.
-5098	The Directory table contains a circular reference.	Check the Knowledge Base for information about this error, or request technical support.
-5097	Could not write custom records to the _Validation table.	Check the Knowledge Base for information about this error, or request technical support.
-5096	Could not retrieve the standard table list.	Check the Knowledge Base for information about this error, or request technical support.
-5095	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5094	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5093	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5092	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5091	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5090	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5089	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5088	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5087	Stop at first error.	The user chose the "Stop build process when first error is encountered" in Tools | Options menu. An error occurred so the build was stopped.
-5086	Treat warnings as errors.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-5085	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5084	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5083	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5082	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5081	Could not write to setup.ini.	Check the Knowledge Base for information about this error, or request technical support.
-5080	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5079	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5078	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5077	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5076	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5075	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5074	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5072	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5071	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5070	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5069	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-5068	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5066	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5065	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5064	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5063	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5062	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5061	The specified filename already exists.	Use the feature's Source Location property to prevent this error.
-5060	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5059	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5058	Could not obtain merge module dependencies.	Check the Knowledge Base for information about this error, or request technical support.
-5057	Could not obtain merge module catalog.	Check the Knowledge Base for information about this error, or request technical support.
-5056	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5054	Could not determine the size of the file "%1".	Verify the file specified exists.
-5053	The file "%1" could not be found.	Verify the file specified exists.
-5052	Could not determine the free space on the volume %1.	Check the Knowledge Base for information about this error, or request technical support.
-5051	Could not remove the read-only attribute from the file "%1".	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-5050	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5049	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5048	Could not create the file "%1".	Check the Knowledge Base for information about this error, or request technical support.
-5047	Cannot create directory %1.	Check the Knowledge Base for information about this error, or request technical support.
-5046	Could not preserve previous Build Reports.	This build error occurs when a release exists, but the corresponding Reports folder in the release folder does not exist. This scenario typically occurs if you manually delete the Reports folder, but not the disk image folder. This scenario may also occur if you upgrade a project from an earlier version of InstallShield. To resolve this error, click the Open Release Folder button. InstallShield opens the current release's folder. Delete all of the files within the folder. Then rebuild the release.
-5045	Could not preserve previous Build Logs.	This build error occurs when a release exists, but the corresponding LogFiles folder in the release folder does not exist. This scenario typically occurs if you manually delete the LogFiles folder, but not the disk image folder. This scenario may also occur if you upgrade a project from an earlier version of InstallShield. To resolve this error, click the Open Release Folder button. InstallShield opens the current release's folder. Delete all of the files within the folder. Then rebuild the release.
-5044	Cannot delete directory %1.	Check the Knowledge Base for information about this error, or request technical support.
-5043	The volume does not exist.	Check in the Releases view to ensure the volume specified exists.
-5042	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5041	The string ID "%1" was used to specify a Feature or Feature destination.	You cannot use a stringID for a destination. Find the feature using this string ID and change the destination using a directory identifier.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-5040	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5039	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5038	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5037	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5036	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5033	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5025	Could not save package.	Check the Knowledge Base for information about this error, or request technical support.
-5024	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5019	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5018	The logical disk does not contain any features.	Check the Knowledge Base for information about this error, or request technical support.
-5017	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5016	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5015	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5014	An error occurred building icon "%1". The specified Icon key was not found in the Icon table.	For information about this error, see article Q105625 in the Knowledge Base.
-5013	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-5011	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5010	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5009	The schema Summary Stream must be 200 or later.	Check the Knowledge Base for information about this error, or request technical support.
-5008	This 32-bit package cannot include 64-bit data. The 64-bit data may come from a merge module.	If your project does not have at least one file, folder, or registry entry that is configured to be installed to a 64-bit location, InstallShield attempts to create a 32-bit .msi package at build time; however, a 32-bit .msi package cannot contain a 64-bit merge module that contains one or more 64-bit components. To resolve this build error, consider the following options: ● Replace the 64-bit merge modules with 32-bit versions. ● Add one or more files, folders, or registry entries to 64-bit locations in your project in order to generate a 64-bit package. Note, however, that 64-bit packages cannot be run on 32-bit target systems. To learn more, see Challenges of Supporting Both 32-Bit and 64-Bit Target Operating Systems.
-5007	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5006	Could not save "%1".	Check the Knowledge Base for information about this error, or request technical support.
-5005	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5004	Could not open the project with write access.	Check to make sure the project is not already open by the IDE.
-5003	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5002	Internal build error.	Check the Knowledge Base for information about this error, or request technical support.
-5001	Could not copy setup.ini.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-4701	An error occurred building the One-Click Install Web page.	Check the Knowledge Base for information about this error, or request technical support.
-4354	The build was unable to extract COM information from a file in a feature.	Ensure that the file is self-registering and verify that the self-registration process does not fail.
-4349	Failed to build %1 information for dynamically extracted COM features.	Check the Knowledge Base for information about this error, or request technical support.
-4348	Failed to build %1 information for dynamically extracted COM features.	Check the Knowledge Base for information about this error, or request technical support.
-4347	Failed to build information for dynamically extracted COM features.	Check the Knowledge Base for information about this error, or request technical support.
-4346	Failed to build information for dynamically extracted COM features.	Check the Knowledge Base for information about this error, or request technical support.
-4345	Failed to build information for dynamically extracted COM features.	Check the Knowledge Base for information about this error, or request technical support.
-4344	Failed to build information for dynamically extracted COM features.	Check the Knowledge Base for information about this error, or request technical support.
-4343	Failed to build information for dynamically extracted COM features.	Check the Knowledge Base for information about this error, or request technical support.
-4340	Failed to initialize COM extraction module.	Check the Knowledge Base for information about this error, or request technical support.
-4303	An unexpected error occurred while synchronizing the file key in the specified feature.	Check the Knowledge Base for information about this error, or request technical support.
-4302	A conflict was encountered while synchronizing file key %1 in feature %2.	Check the Knowledge Base for information about this error, or request technical support.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-4301	Could not find the .msi file for key synchronization.	Check to ensure the specified MSI file exists.
-4092	Error opening MSI database %1.	Check the Knowledge Base for information about this error, or request technical support.
-4075	File not found. An error occurred merging module '%1' for feature '%2'.	Verify that the merge module exists in the merge module search path.
-4072	Error retrieving the dependencies for %1.	Check the Knowledge Base for information about this error, or request technical support.
-4006	Failed to delete all of the files in a directory	Close any open files in the build output directory (Disk1). Close Msidb.exe if it is open.
-3876	Ignoring invalid template %1 in the Summary Stream.	Check the Knowledge Base for information about this error, or request technical support.
-3713	The function block must be in the form "Module::Function".	Change the function block to follow the form "Module::Function".
-3114	Application path %1 does not contain destination of associated feature %2.	Check the Knowledge Base for information about this error, or request technical support. Because an application path is also a registry entry, this error might occur if a registry path like the following one is created, but it is missing the path entry: HKEY_LOCAL_MACHINE\Microsoft\Windows\CurrentVersion\App Paths To correct this, specify a path entry.
-3028	An error occurred replacing string IDs in the specified table.	A required string ID was left blank in the specified table. Provide a string ID.
-3016	Failed to add Binary table %1 to package.	Check the Knowledge Base for information about this error, or request technical support.
-2200	Could not overwrite file.	Ensure the specified file is not read-only.
-1531	The size specified for the disk is too small for the feature.	Using the Release Wizard, change the Media Format to a larger size.
-1530	The size specified for the disk is too small for the file.	Using the Release Wizard, change the Media Format to a larger size.

Table 9-2 ▪ Build Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-1527	No files are included in the project.	This warning occurs when you have not included any files in your project. Use the Files view to add files into your project to avoid this error.
-1505	Could not add the CAB file to the MSI package.	Check the Knowledge Base for information about this error, or request technical support.
-1501	Could not compress "%1" into "%2".	Ensure that IsCmdBld.exe is executed in the relative directory. You can do this by locating the files specified in the error message and launching IsCmdBld.exe within their directory.
-1027	Failed signing %1.	Check the digital signature information (digital keys and password) provided for the current release.
-1024	File not found. Cannot stream the file into the Binary table.	Ensure the specified file exists.
-1014	Cannot rename a directory.	Windows Explorer or a DOS prompt may be pointing to a subfolder of the release output folder (Disk1) or to the Interm folder, locking it. Change the current directory. Close any open files in the Disk1 folder. Close Msidb.exe if it is open.
-1013	The specified file is being used by another program.	Close the application that is currently using the file and re-run the build process.
-1009	Insufficient disk space or the target drive cannot be located.	Increase disk space on the build target, or select a new target for the build. If the target drive cannot be located, select a new target, or ensure permissions are set correctly for the target drive.
-1001	Error opening MSI database	Your MSI engine may be corrupt. You need to repair or reinstall the engine. For further information, check article Q105892 in the Knowledge Base for information about this error, or request technical support.
-1000	Invalid product configuration. Failed to create directory.	Check the Knowledge Base for information about this error, or request technical support.

Upgrade Errors and Warnings

The following table describes potential errors and warnings that may occur when you upgrade a project from an earlier version of InstallShield to the current version. It also provides tips to correct the issues.

Table 9-3 ▪ Upgrade Errors and Warnings

Error	Description	Troubleshooting Tips
-72	Config.sys no longer supported.	Changes to Config.sys are no longer supported. These changes are not migrated. ***Note ▪*** *There is an example of a custom action to modify Config.sys in the `InstallShield Program Files Folder`\Samples folder.*
-73	Autoexec.bat no longer supported.	Changes to `Autoexec.bat` are no longer supported. These changes are not migrated. ***Note ▪*** *You can modify environment variables from the Environment Variables view.*
-74	Uninstall Icon no longer supported.	For more information, see Creating an Uninstallation Shortcut.
-75	Splash screen no longer supported.	The Splash screen has been replaced by the Splash dialog.
-77	Logo bitmap no longer supported.	This aspect of your setup is not migrated.
-78	Background color no longer supported.	This aspect of your setup is not migrated.
-248	Filegroup %1 had Windows NT 3.51 marked as a Target OS. Migrating to NT4.	The specified file group could not be migrated as an NT 3.51 feature because the Windows Installer architecture does not support that operating system. The new feature created for you is marked for Windows NT4.
-249	Target directory %1 could not be directly migrated. Migrating to destination folder %2 in feature %3.	The specified directory could not be migrated as one cohesive piece. The destination folder specified now contains contents of the migrated target directory.

Table 9-3 ▪ Upgrade Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-250	Converted illegal destination <SRCDIR> of file group to [INSTALLDIR].	INSTALLDIR is the default destination folder for all of your setup's files.
-251	<SRCDISK> is not supported. Converting destination to [INSTALLDIR].	SRCDISK is not supported in the new InstallShield. INSTALLDIR is the default destination folder for all of your setup's files.
-405	Registry hive %1 not supported. Ignoring this key.	The specified registry hive is not supported by Windows Installer. Therefore, the information contained within that hive was not migrated.
-406	Registry type %1 not supported. Ignoring this entry.	The specified registry type is not supported by Windows Installer. Therefore, the information contained within that entry was not migrated.
-407	This registry entry was not marked to be uninstalled in your old project. It will be uninstalled in the new version.	The specified registry entry will be uninstalled with the rest of your product during uninstallation.
-535	Reboot option not migrated; use the Custom Actions view to add a reboot action.	Rebooting the system during installation is handled via custom actions. For more information, see Custom Action Gallery.
-536	Individual file group selection no longer supported.	Since file groups no longer exist, this setting cannot be migrated.
-537	Welcome dialog is required	This mandatory dialog has been added to your setup.
-538	SetupComplete dialog is required	This mandatory dialog has been added to your setup.
-541	Temp Files not currently supported.	The use of temporary files is not supported by InstallShield. These files are not migrated. Use the Setup Files view to add files you want available on the target system only during installation.
-542	Disk1 Files no longer supported.	These changes are not migrated.

Table 9-3 ▪ Upgrade Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
-544	The LicenseAgreement dialog .txt file was converted to an .rtf file for use in Express 3.0.	InstallShield requires rich text (.rtf) files for the LicenseAgreement dialog. Your text (.txt) file was converted to .rtf by InstallShield.
-545	The ReadMeFileBrowser dialog .txt file was converted to an .rtf file for use in Express 3.0.	InstallShield requires rich text (.rtf) files for the ReadMeFileBrowser dialog. Your text (.txt) file was converted to .rtf by InstallShield.
-601	Could not migrate extension %1.	The specified extension could not be migrated. Re-create this extension in the File Extensions view.
-602	Used first found file %1 as the binary file for extension %2.	The specified file was migrated for your extension's binary.

Windows Installer Run-time Errors

The following table contains a listing of Windows Installer errors that may occur during run time of an installation. The errors are related to the built-in InstallShield custom actions that are added automatically to InstallShield projects to support different functionality. For a list of the custom actions, see InstallShield Custom Action Reference.

Tip ▪ *You can find detailed information—including resolution information—on some of the run-time errors in the Knowledge Base.*

Table 9-4 ▪ Windows Installer Run-time Errors

Error Number	Message	Troubleshooting Information
27500	This setup requires Internet Information Server for configuring IIS Virtual Roots. Please make sure that you have IIS installed.	This error may occur if your project contains IIS data that you configured in the Internet Information Services view. If you encounter this error, ensure that the target system has IIS installed.

Table 9-4 ▪ Windows Installer Run-time Errors (cont.)

Error Number	Message	Troubleshooting Information
27501	This setup requires Administrator privileges for configuring IIS Virtual Roots.	This error may occur if your project contains IIS data that you configured in the Internet Information Services view. This error occurs if you do not have administrator privileges for configuring IIS virtual directories, but you are running an installation that configures IIS virtual directories.
27508	Error installing COM+ application [2]. [3]	This error may occur if your project contains a COM+ application that you configured in the Component Services view. This error occurs if there are missing dependencies for COM+ component DLLs or incorrect settings for your COM+ application. To resolve this error, add the dependencies to your project or check the COM+ application in your project and make any necessary changes.
27509	Error uninstalling COM+ application [2]. [3]	This error may occur if your project contains a COM+ application that you configured in the Component Services view. This error occurs if the run time fails to delete the COM+ application from the system.
27510	Error installing COM+ application [2]. Could not load Microsoft(R) .NET class libraries. Registering .NET serviced components requires that Microsoft(R) .NET Framework be installed.	This error may occur if your project contains a COM+ application that you configured in the Component Services view. To prevent this error from occurring, consider adding the .NET Framework to your project. To learn how, see Adding .NET Framework Redistributables to Projects. As an alternative, you can use the Installation Requirements page in the Project Assistant to add a .NET Framework requirement to your project. If a target system does not have the appropriate version of the .NET Framework, the installation displays an error message, and the product is not installed.

Table 9-4 ▪ Windows Installer Run-time Errors (cont.)

Error Number	Message	Troubleshooting Information
27517	This installation requires Administrator privileges for installing COM+ applications. Log on as an administrator and then retry this installation.	This error may occur if your project contains a COM+ application that you configured in the Component Services view. This error occurs if you do not have administrator privileges for installing COM+ applications, but you are running an installation that includes COM+ applications.
27555	Error attempting to apply permissions to object [2]. System error: [3] ([4])	This error may occur if you used the custom InstallShield handling for securing files, folders, or registry keys in your project. To learn more about this functionality, see Securing Files, Folders, and Registry Keys in a Locked-Down Environment.

Setup.exe Run-Time Errors and Warnings

The table below lists the errors that you may encounter when running `Setup.exe`. You can trap these return values when you call **CreateProcess()** to launch `Setup.exe`. However, `Setup.exe` does not return error codes from `MsiExec.exe`.

Note ▪ *If you encounter an error, the strings displayed may not be displayed in English if your operating system is not running in English.*

Table 9-5 ▪ Setup.exe Run-Time Errors and Warnings

Error	Description	Troubleshooting Tips
-1	The user canceled the setup.	Run the setup again without pressing the Esc key or clicking the Cancel button.
1150	Setup has detected an incompatible version of Windows. Please click OK and relaunch setup on Windows 95, Windows NT 4.0, or later.	Windows Installer is compatible with NT 4.0 and later, and Windows 9x. Check your version of Windows and upgrade if necessary.
1151	Error writing to the temporary location.	In order to write to the temporary location, the environment variable TEMP must be set. Check to see that Temp folder exists and has enough disk space to accommodate the setup. If there are files in the Temp folder, delete them and rerun Setup.exe.

Table 9-5 ▪ Setup.exe Run-Time Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
1152	Error extracting <file name> to the temporary location.	Check to see that you are able to write to the Temp folder (see errors above). If the Temp folder is valid, there may be corrupt files in the setup. Check the files to ensure none are corrupted and rerun Setup.exe.
1153	Error reading setup initialization file.	The Setup.ini file must be located in the same folder as Setup.exe. If not, move Setup.ini to that location.
1154	Installer not found in <path>.	Windows Installer may not have been properly installed, or you may have an older version. Reinstall if necessary.
1155	File <file name> not found.	Make sure the .msi file exists. If so, make sure it is located in the same folder as Setup.exe.
1157	Failed to launch MsiExec.exe.	Make sure you are distributing the correct version or versions of Windows Installer for the target platform. Check the syntax on your MsiExec.exe command-line arguments.
1158	Error populating strings.	Verify all strings in Setup.ini are valid. Refer to the InstallShield License Agreement for more information.
1201	Setup needs <amount> KB free space in <folder>. Please free up some space and try again.	There is insufficient disk space in your target location. Please make sure there is at least 10 MB of free space in the drive where the setup is set to install.
1202	You do not have sufficient privileges to complete this installation for all users of the machine. Log on as an administrator and then retry this installation.	You must have administrative rights to complete this installation.
1208	ANSI code page for <language> is not installed on the system and therefore setup cannot run in the selected language. Run the setup and select another language.	Run the setup and select another language.

Table 9-5 ▪ Setup.exe Run-Time Errors and Warnings (cont.)

Error	Description	Troubleshooting Tips
1603	General Windows Installer engine error. Increase DiskSpace requirement in Setup.ini and try again.	Reinstall Windows Installer by running InstMsiW.exe file (for Windows NT and 2000) or InstMsiA.exe (for Windows 9x) to reinstall.
1614	An error occurred while downloading a file.	Check connectivity and disk space on the target machine and retry the download. Ensure Internet Explorer 3.02 or later is installed on the target system.
1627	Unable to save file.	The specified file was unable to be saved. This can be for a variety of reasons, including insufficient permissions on the target system, loss of connectivity, lack of space, etc. Ensure you have access to the target directory.

Visual Studio Project Import Errors and Warnings

This table lists the errors and warnings that might occur when you do either of the following:

- Convert a Visual Studio setup project (.vdproj) to an InstallShield project (.ise).

- Import a Visual Studio setup or merge module project (.vdproj) into an InstallShield project (.ise).

Table 9-6 ▪ Visual Studio Project Import Errors and Warnings

Error or Warning Number	Description	Troubleshooting Tips
-9000	An unknown exception occurred.	Check the Knowledge Base for information about this error, or request technical support.
-9001	An unknown COM exception occurred.	Check the Knowledge Base for information about this error, or request technical support.
-9002	An error occurred while loading the project %1.	This error may occur if the Visual Studio setup or merge module project file is corrupted or if InstallShield cannot correctly read the Visual Studio setup project file. To resolve this error, request technical support.
-9003	An error occurred while creating the project %1.	This error occurs if InstallShield cannot create an Express project from the Visual Studio setup project. InstallShield tries to create the project in the same folder that contains the Visual Studio project file (.vdproj). To resolve this error, see if the folder that contains the Visual Studio project file is read-only, and change it to writable.
-9004	An error occurred while saving the project %1.	This error occurs if InstallShield cannot create an Express project from the Visual Studio setup project. InstallShield tries to save the project in the same folder that contains the Visual Studio project file (.vdproj). To resolve this error, see if the folder that contains the Visual Studio project file is read-only, and change it to writable.
-9005	An error occurred while converting the project.	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support
-9006	An error occurred while determining the project type.	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support
-9007	An error occurred while parsing the section '%1'.	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support

Table 9-6 ▪ Visual Studio Project Import Errors and Warnings (cont.)

Error or Warning Number	Description	Troubleshooting Tips
-9008	No open brace found for the section %1 under %2.	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support
-9009	No close brace found for the section %1 under %2.	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support
-9010	An invalid entry %1 found in the section %1 under %2.	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support
-9011	An error occurred while adding the property %1 to the section %2.	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support.
-9012	An error occurred while adding the section %1 to the section %2.	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support.
-9013	The section %1 does not exist.	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support.
-9014	An error occurred while converting the product properties.	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support.
-9015	An error occurred while converting the files.	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support.
-9016	The file type %1 is invalid. The file %2 was not converted.	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support.

Table 9-6 ▪ Visual Studio Project Import Errors and Warnings (cont.)

Error or Warning Number	Description	Troubleshooting Tips
-9017	An error occurred while converting the file %1. Type: %2	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support.
-9018	An error occurred while converting the features.	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support.
-9019	An error occurred while converting the feature %1.	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support.
-9020	An error occurred while converting the folders.	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support.
-9021	An error occurred while converting the folder %1. Type: %2	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support.
-9022	The folder type %1 is invalid. The folder %2 was not converted.	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support.
-9023	An error occurred while converting the custom actions.	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support.
-9024	An error occurred while converting the custom action %1. Type: %2	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support.
-9025	The custom action type %1 is invalid. The custom action %2 was not converted.	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support.

Table 9-6 ▪ Visual Studio Project Import Errors and Warnings (cont.)

Error or Warning Number	Description	Troubleshooting Tips
-9026	A component for the source file %1 of the custom action %2 was not found.	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support.
-9027	An error occurred while converting the file types.	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support.
-9028	An error occurred while converting the file extension %1. Type: %2	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support.
-9029	The file extension type %1 is invalid. The file extension %2 was not converted.	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support.
-9030	An error occurred while converting the verb %1. Type: %2	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support.
-9031	The verb type %1 is invalid. The verb %2 was not converted.	This error may occur if the Visual Studio project file is corrupted or if InstallShield cannot correctly read the Visual Studio project file. To resolve this error, request technical support.
-9032	A command is not specified for the file extension %1. The file extension was not converted.	This warning occurs if the Command property for the specified file type was left blank in the File Types Editor in Visual Studio. The warning alerts you that InstallShield could not configure the file extension during the conversion process. To resolve this issue, specify a command for the file type in Visual Studio, and then convert the Visual Studio setup project to an InstallShield project. Otherwise, you can add and configure the file extension in the File Extensions view in InstallShield.

Table 9-6 ▪ Visual Studio Project Import Errors and Warnings (cont.)

Error or Warning Number	Description	Troubleshooting Tips
-9033	A component for the command %1 of the file extension %2 was not found. The file extension was not converted.	This error may occur if the Visual Studio setup project file is corrupted or if InstallShield cannot correctly read the Visual Studio setup project file. To resolve this error, request technical support.
-9034	An error occurred while converting the registries.	This error may occur if the Visual Studio setup project file is corrupted or if InstallShield cannot correctly read the Visual Studio setup project file. To resolve this error, request technical support.
-9035	An error occurred while converting the registry key %1. Type: %2	This error may occur if the Visual Studio setup project file is corrupted or if InstallShield cannot correctly read the Visual Studio setup project file. To resolve this error, request technical support.
-9036	The registry key type %1 is invalid. The registry key %2 was not converted.	This error may occur if the Visual Studio setup project file is corrupted or if InstallShield cannot correctly read the Visual Studio setup project file. To resolve this error, request technical support.
-9037	An error occurred while converting the registry value %1. Type: %2	This error may occur if the Visual Studio setup project file is corrupted or if InstallShield cannot correctly read the Visual Studio setup project file. To resolve this error, request technical support.
-9038	The registry value type %1 is invalid. The registry value %2 was not converted.	This error may occur if the Visual Studio setup project file is corrupted or if InstallShield cannot correctly read the Visual Studio setup project file. To resolve this error, request technical support.
-9039	An error occurred while converting the shortcuts.	This error may occur if the Visual Studio setup project file is corrupted or if InstallShield cannot correctly read the Visual Studio setup project file. To resolve this error, request technical support.
-9040	An error occurred while converting the shortcut %1. Type: %2	This error may occur if the Visual Studio setup project file is corrupted or if InstallShield cannot correctly read the Visual Studio setup project file. To resolve this error, request technical support.
-9041	The shortcut type %1 is invalid. The shortcut %2 was not converted.	This error may occur if the Visual Studio setup project file is corrupted or if InstallShield cannot correctly read the Visual Studio setup project file. To resolve this error, request technical support.

Table 9-6 ▪ Visual Studio Project Import Errors and Warnings (cont.)

Error or Warning Number	Description	Troubleshooting Tips
-9042	The shortcut target %1 is invalid. The shortcut %2 was not converted.	This error may occur if the Visual Studio setup project file is corrupted or if InstallShield cannot correctly read the Visual Studio setup project file. To resolve this error, request technical support.
-9043	An error occurred while converting the launch conditions.	This error may occur if the Visual Studio setup project file is corrupted or if InstallShield cannot correctly read the Visual Studio setup project file. To resolve this error, request technical support.
-9044	An error occurred while converting the launch condition %1. Type: %2	This error may occur if the Visual Studio setup project file is corrupted or if InstallShield cannot correctly read the Visual Studio setup project file. To resolve this error, request technical support.
-9045	The launch condition type %1 is invalid. The launch condition %2 was not converted.	This error may occur if the Visual Studio setup project file is corrupted or if InstallShield cannot correctly read the Visual Studio setup project file. To resolve this error, request technical support.
-9046	An error occurred while converting the locators.	This error may occur if the Visual Studio setup project file is corrupted or if InstallShield cannot correctly read the Visual Studio setup project file. To resolve this error, request technical support.
-9047	An error occurred while converting the locator %1. Type: %2	This error may occur if the Visual Studio setup project file is corrupted or if InstallShield cannot correctly read the Visual Studio setup project file. To resolve this error, request technical support.
-9048	The locator type %1 is invalid. The locator %2 was not converted.	This error may occur if the Visual Studio setup project file is corrupted or if InstallShield cannot correctly read the Visual Studio setup project file. To resolve this error, request technical support.
-9049	An error occurred while converting the product icon.	This error may occur if the Visual Studio setup project file is corrupted or if InstallShield cannot correctly read the Visual Studio setup project file. To resolve this error, request technical support.
-9050	The project language '%1' is not installed in InstallShield. The language was not converted.	This warning occurs if you convert a Visual Studio project that includes support for a language, but InstallShield does not have built-in support for that language.

Table 9-6 ▪ Visual Studio Project Import Errors and Warnings (cont.)

Error or Warning Number	Description	Troubleshooting Tips
-9052	File type '%1' specifies no extensions. The file type was not converted.	This warning occurs if the Extensions property for the specified file type was left blank in the File Types Editor in Visual Studio. The warning alerts you that InstallShield could not configure the file extension during the import or conversion process. To resolve this issue, specify one or more file extensions for the file type in Visual Studio, and then convert or import the Visual Studio project into an InstallShield project. Otherwise, you can add and configure the file extension in the File Extensions view in InstallShield after you have converted or imported the Visual Studio project.
-9053	Launch condition '%1' specifies no conditions. The launch condition was not converted.	This warning occurs if the Condition property for the specified launch condition was left blank in the Launch Conditions Editor in Visual Studio. The warning alerts you that InstallShield could not configure the launch condition during the import or conversion process. To resolve this issue, specify a condition in Visual Studio, and then convert or import the Visual Studio project into an InstallShield project. Otherwise, you can add and configure the launch condition in the Requirements view in InstallShield after you have converted or imported the Visual Studio project.
-9054	File search '%1' specifies no file names. The file search was not converted.	This warning occurs if the FileName property for the specified file search was left blank in the Launch Conditions Editor in Visual Studio. The warning alerts you that InstallShield could not configure the file search during the import or conversion process. To resolve this issue, specify the file name for the file search in Visual Studio, and then convert or import the Visual Studio project into an InstallShield project. Otherwise, you can add and configure the file search in the Requirements view in InstallShield after you have converted or imported the Visual Studio project.

Table 9-6 ▪ Visual Studio Project Import Errors and Warnings (cont.)

Error or Warning Number	Description	Troubleshooting Tips
-9055	Feature '%1' already exists in the InstallShield project. The feature was not converted.	 ***Edition ▪*** *This warning may occur in the InstallShield Premier and Professional Editions of InstallShield.* *This warning is not applicable to the Express edition; in this edition, InstallShield adds all of the application data from the Visual Studio project to the Always Install feature in the InstallShield project.* This warning occurs if you have a Visual Studio project that contains a particular feature and you import that project into an InstallShield project that already contains that feature. This scenario may occur if you import the same Visual Studio project into your InstallShield project more than once.
-9056	Directory '%1' already exists in the InstallShield project. The folder '%2' was not converted.	This warning occurs if you have a Visual Studio project that contains a particular folder and you import that project into an InstallShield project that already contains that folder. This scenario may occur if you import the same Visual Studio project into your InstallShield project more than once. If you encounter this warning, check the Files view to ensure that the folder that is identified in the warning message is not missing from your InstallShield project.
-9057	File key '%1' already exists in the InstallShield project. The file '%2' was not converted.	This warning occurs if you have a Visual Studio project that contains a file that is associated with a particular file key and you import that project into an InstallShield project that already contains a file with that same file key. This scenario may occur if you import the same Visual Studio project into your InstallShield project more than once. If you encounter this warning, check the Files view to ensure that the file that is identified in the warning message is not missing from your InstallShield project. ***Edition ▪*** *The Premier and Professional editions of InstallShield contain a Direct Editor view that shows the file key that is associated with each file in your project. The file keys are the primary keys of the* `File` *table in the .msi database; the* `File` *table cannot contain duplicate file keys.* *You can also see file key information in the Files area of the Components view in the Premier and Professional editions of InstallShield.*

Table 9-6 ▪ Visual Studio Project Import Errors and Warnings (cont.)

Error or Warning Number	Description	Troubleshooting Tips
-9058	Shortcut key '%1' already exists in the InstallShield project. The shortcut '%2' was not converted.	This warning occurs if you have a Visual Studio project that contains a shortcut that is associated with a particular shortcut key and you import that project into an InstallShield project that already contains a shortcut with that same shortcut key. This scenario may occur if you import the same Visual Studio project into your InstallShield project more than once. If you encounter this warning, check the Shortcuts/Folders view to ensure that the shortcut that is identified in the warning message is not missing from your InstallShield project. ***Edition ▪*** *The Premier and Professional editions of InstallShield contain a Direct Editor view that shows the shortcut key that is associated with each shortcut in your project. The shortcut keys are the primary keys of the* `Shortcut` *table in the .msi database; the* `Shortcut` *table cannot contain duplicate shortcut keys.*
-9059	Extension '%1' already exists in the InstallShield project. The file type '%2' was not converted.	This warning occurs if you have a Visual Studio project that contains a particular file extension and you import that project into an InstallShield project that already contains that same file extension. This scenario may occur if you import the same Visual Studio project into your InstallShield project more than once. If you encounter this warning, check the File Extensions view to ensure that the file extension that is identified in the warning message is not missing from your InstallShield project.
-9060	Registry key '%1' already exists in the InstallShield project. The registry entry 'Key: %2 Value: %3 Data: %4' was not converted.	This warning occurs if you have a Visual Studio project that contains a particular registry entry and you import that project into an InstallShield project that already contains that same registry entry. This scenario may occur if you import the same Visual Studio project into your InstallShield project more than once. If you encounter this warning, check the Registry view to ensure that the registry entry that is identified in the warning message is not missing from your InstallShield project.

Table 9-6 ▪ Visual Studio Project Import Errors and Warnings (cont.)

Error or Warning Number	Description	Troubleshooting Tips
-9061	Custom action '%1' already exists in the InstallShield project. The custom action named '%2' was not converted.	This warning occurs if you have a Visual Studio project that contains a particular custom action and you import that project into an InstallShield project that already contains a custom action with that same name. This scenario may occur if you import the same Visual Studio project into your InstallShield project more than once. If you encounter this warning, check the Custom Actions view to ensure that the custom action that is identified in the warning message is not missing from your InstallShield project.
-9062	Launch condition '%1' already exists in the InstallShield project. The launch condition named '%2' was not converted.	This warning occurs if you have a Visual Studio project that contains a particular launch condition and you import that project into an InstallShield project that already contains a launch condition with that same name. This scenario may occur if you import the same Visual Studio project into your InstallShield project more than once. If you encounter this warning, check the Requirements view to ensure that the launch condition that is identified in the warning message is not missing from your InstallShield project.
-9063	RegLocator '%1' already exists in the InstallShield project. The registry search '%2' was not converted.	This warning occurs if you have a Visual Studio project that contains a particular registry search and you import that project into an InstallShield project that already contains a registry search with that same name. This scenario may occur if you import the same Visual Studio project into your InstallShield project more than once. If you encounter this warning, check the Requirements view to ensure that the registry search that is identified in the warning message is not missing from your InstallShield project.
-9064	DrLocator '%1' already exists in the InstallShield project. The file search '%2' was not converted.	This warning occurs if you have a Visual Studio project that contains a particular file or folder search and you import that project into an InstallShield project that already contains a file or folder search with that same name. This scenario may occur if you import the same Visual Studio project into your InstallShield project more than once. If you encounter this warning, check the Requirements view to ensure that the file or folder search that is identified in the warning message is not missing from your InstallShield project.

Table 9-6 ▪ Visual Studio Project Import Errors and Warnings (cont.)

Error or Warning Number	Description	Troubleshooting Tips
-9065	CompLocator '%1' already exists in the InstallShield project. The Windows Installer search '%2' was not converted.	This warning occurs if you have a Visual Studio project that contains a particular file or folder search and you import that project into an InstallShield project that already contains a file or folder search with that same name. This scenario may occur if you import the same Visual Studio project into your InstallShield project more than once. If you encounter this warning, check the Requirements view to ensure that the file or folder search that is identified in the warning message is not missing from your InstallShield project.
-9069	Windows Installer properties that resolve to 64-bit locations are not supported in this edition of InstallShield. The folder '%1' was converted to '%2'.	 ***Edition ▪*** *Support for Windows Installer properties that resolve to 64-bit locations is available in the following editions of InstallShield:* • *Premier* • *Professional* This warning occurs if the DefaultLocation property in your Visual Studio project uses a 64-bit location (**[ProgramFiles64Folder]**, **[CommonFiles64Folder]**, or **[System64Folder]**) and if you indicate in the Visual Studio Deployment Project Import Wizard that you want to import the INSTALLDIR value from your Visual Studio project. If this warning occurs, InstallShield Express Edition uses the 32-bit equivalent of the folder (for example, **[ProgramFilesFolder]**). The Premier and Professional editions use the 64-bit location. If you require 64-bit support in your installation, consider upgrading to the Premier or Professional editions of InstallShield; these editions enable you to create 64-bit packages, install to 64-bit locations, and configure other 64-bit related settings.

Table 9-6 ▪ Visual Studio Project Import Errors and Warnings (cont.)

Error or Warning Number	Description	Troubleshooting Tips
-9070	An error occurred while converting the prerequisites.	 ***Edition ▪*** *InstallShield Premier Edition and InstallShield Professional Edition include the InstallShield Prerequisite Editor, a tool that enables you to define custom InstallShield prerequisites and to edit settings for any existing InstallShield prerequisites. If you have either one of these editions of InstallShield, you would be able to create your own InstallShield prerequisites that would install the Microsoft redistributables during your product's installation.* This error occurs if you import a Visual Studio setup project that contains a prerequisite but InstallShield encounters a problem when trying to map the Visual Studio prerequisite to an equivalent InstallShield prerequisite. To resolve this error, request technical support.
-9071	InstallShield has no equivalent InstallShield prerequisite for '%1'. The prerequisite was not converted.	 ***Edition ▪*** *InstallShield Premier Edition and InstallShield Professional Edition include the InstallShield Prerequisite Editor, a tool that enables you to define custom InstallShield prerequisites and to edit settings for any existing InstallShield prerequisites. If you have either one of these editions of InstallShield, you would be able to create your own InstallShield prerequisites that would install the Microsoft redistributables during your product's installation.* This warning occurs if you import a Visual Studio setup project that contains a prerequisite but InstallShield does not have an equivalent InstallShield prerequisite. To resolve this warning, consider upgrading to the Premier or Professional edition of InstallShield; these editions enable you to create your own InstallShield prerequisites and add them to your InstallShield installation projects.

Table 9-6 ▪ Visual Studio Project Import Errors and Warnings (cont.)

Error or Warning Number	Description	Troubleshooting Tips
-9072	Launch conditions are not supported in this edition of InstallShield. The launch condition named '%1' was not converted.	 ***Edition ▪*** *The following editions let you create launch conditions.* • *Premier* • *Professional* *If one or more of launch conditions are false on the target system, the installation exits and an error message is displayed.* This warning occurs if your Visual Studio project contains a launch condition and you try to import that project into an InstallShield project. To resolve this warning, consider upgrading to the Premier or Professional editions of InstallShield.
-9073	Windows Installer searches are not supported in this edition of InstallShield. The Windows Installer search named '%1' was not converted.	 ***Edition ▪*** *The System Search Wizard in the following editions of InstallShield let you create Windows Installer searches.* • *Premier* • *Professional* This warning occurs if your Visual Studio project contains a Windows Installer search for the existence of a particular component ID on the target system, and you try to import that project into an InstallShield project. To resolve this warning, consider upgrading to the Premier or Professional edition of InstallShield, which have support for this type of target system search.
-9074	An error occurred while converting the project outputs.	This error occurs if you convert a Visual Studio setup or merge module project that contains one or more project outputs but InstallShield encounters a problem when incorporating the project outputs into the InstallShield project. To resolve this error, request technical support.

Table 9-6 ▪ Visual Studio Project Import Errors and Warnings (cont.)

Error or Warning Number	Description	Troubleshooting Tips
-9075	An error occurred while converting the project output %1. Type: %2	This error occurs if you convert a Visual Studio setup or merge module project that contains one or more project outputs but InstallShield encounters a problem when incorporating the project outputs into the InstallShield project. To resolve this error, request technical support.
-9076	The project output type %1 is invalid. The project output %2 was not converted.	This error occurs if you convert a Visual Studio setup or merge module project that contains one or more project outputs but one of the project outputs is invalid. To resolve this error, review the project output in your Visual Studio project, and try to resolve any issues with it. Then convert the Visual Studio project to an InstallShield project.
-9077	File key '%1' already exists in the InstallShield project. The project output '%2' was not converted.	This warning occurs if you have a Visual Studio project that contains a project output that has a file that is associated with a particular file key, and you import that project into an InstallShield project that already contains a file with that same file key. This scenario may occur if you import the same Visual Studio project into your InstallShield project more than once. If you encounter this warning, check the Files view to ensure that the file that is identified in the warning message is not missing from your InstallShield project. ***Edition ▪*** *The Premier and Professional editions of InstallShield contain a Direct Editor view that shows the file key that is associated with each file in your project. The file keys are the primary keys of the* `File` *table in the .msi database; the* `File` *table cannot contain duplicate file keys.* *You can also see file key information in the Files area of the Components view in the Premier and Professional editions of InstallShield.*
-9078	The InstallShield project is not in a Visual Studio solution. The project output '%1' was not converted.	This error occurs if your InstallShield project is open in InstallShield, but not from within Visual Studio, and you try to import a Visual Studio project that contains project outputs into the InstallShield project. To resolve this error, open your InstallShield project from within Visual Studio, and then import the Visual Studio project into it.

Table 9-6 ▪ Visual Studio Project Import Errors and Warnings (cont.)

Error or Warning Number	Description	Troubleshooting Tips
-9079	The Visual Studio project '%1' could not be found in the current Visual Studio solution. The project output '%2' was not converted.	If you want to import into an InstallShield project a Visual Studio setup or merge module project that contains a project output, the Visual Studio project must be in the same solution as all of its project dependencies. Otherwise, this error occurs when you try to import the Visual Studio project. To resolve this error, ensure that your InstallShield project is in the same Visual Studio solution that contains the Visual Studio setup or merge module project that you are importing. In addition, ensure that the solution also includes all of the other Visual Studio projects that are dependencies of the Visual Studio setup or merge module project. Then you can import the Visual Studio setup or merge module project into your InstallShield project.
-9080	The project output group '%1' could not be found in the Visual Studio project '%2'. The project output '%3' was not converted.	This error occurs if you convert a Visual Studio setup or merge module project that contains a project output group but InstallShield encounters a problem when incorporating a project output of that group into the InstallShield project. To resolve this error, request technical support.
-9081	File '%1' is specified to be excluded from a setup or merge module. The file was not converted.	This warning occurs if you import into an InstallShield project a Visual Studio setup or merge module project that contains a file whose Exclude property is set to True. If you want the specified file to be excluded from your InstallShield project, you can ignore this warning. If you want the specified file to be included in your InstallShield project, you can use the Files view in InstallShield to add it. For more information, see Adding Files and Folders to a Project.
-9082	Project output '%1' is specified to be excluded from a setup or merge module. The project output was not converted.	This warning occurs if you import into an InstallShield project a Visual Studio setup or merge module project that contains a file in a project output, and True is selected for the Exclude property of that file. If you want the specified file to be excluded from your InstallShield project, you can ignore this warning. If you want the specified file to be included in your InstallShield project, you can use the Files view in InstallShield to add it. For more information, see Adding References to Visual Studio Solutions.

Table 9-6 ▪ Visual Studio Project Import Errors and Warnings (cont.)

Error or Warning Number	Description	Troubleshooting Tips
-9083	Visual Studio default launch conditions are not supported in InstallShield. The launch condition named '%1' was not converted.	This warning occurs if you import into InstallShield project a Visual Studio setup or merge module project that contains a default launch condition. If you want the specified launch condition to be excluded from your InstallShield project, you can ignore this warning. If you want to add a launch condition to your InstallShield project to check the target system for the default launch condition that Visual Studio added to the original project, you can use the Requirements view in InstallShield to add it. For more information, see Specifying Software Requirements for Your Product.
-9084	Windows Installer properties that resolve to 64-bit locations are not supported in this edition of InstallShield. The Folder property '%1' for the file search named '%2' was converted to '%3'	 ***Edition ▪*** *Support for Windows Installer properties that resolve to 64-bit locations is available in the following editions of InstallShield:* • *Premier* • *Professional* This warning occurs if the Folder property for the specified file search in the Launch Conditions Editor in Visual Studio contains a 64-bit folder location. If this warning occurs, InstallShield Express Edition uses the 32-bit equivalent of the folder (for example, **[ProgramFilesFolder]**). The Premier and Professional editions use the 64-bit location. If you require 64-bit support in your installation, consider upgrading to the Premier or Professional editions of InstallShield; these editions enable you to create 64-bit packages, install to 64-bit locations, and configure other 64-bit related settings.

Table 9-6 ▪ Visual Studio Project Import Errors and Warnings (cont.)

Error or Warning Number	Description	Troubleshooting Tips
-9085	Condition '%1' for the custom action named '%2' is not supported in this edition of InstallShield. The Condition property was not converted.	 ***Edition ▪*** *The following editions of InstallShield include support for creating advanced conditions for custom actions:* • *Premier* • *Professional* This warning occurs if you try to import a Visual Studio project into an InstallShield project, but the Visual Studio project contains a custom action that has a type of condition that is not supported in edition of InstallShield that you are using. To resolve this warning, use the Custom Actions view in InstallShield to add a condition to the custom action after you have converted or imported the Visual Studio project.
-9086	MinDate '%1' for the file search named '%2' is not valid. The MinDate property was not converted.	This warning occurs if you attempt to import or convert a Visual Studio setup that has a file search with an invalid value for the MinDate property. To resolve this warning: 1. Open the **Requirements** view in InstallShield. 2. Right-click the file search and then click **Modify Launch Condition**. The **System Search Wizard** opens. 3. On the second panel of this wizard, click the **Details** button. The **File Details** dialog box opens. 4. Enter the appropriate details.
-9087	MaxDate '%1' for the file search named '%2' is not valid. The MaxDate property was not converted.	This warning occurs if you attempt to import or convert a Visual Studio setup that has a file search with an invalid value for the MaxDate property. To resolve this warning: 1. Open the **Requirements** view in InstallShield. 2. Right-click the file search and then click **Modify Launch Condition**. The **System Search Wizard** opens. 3. On the second panel of this wizard, click the **Details** button. The **File Details** dialog box opens. 4. Enter the appropriate details.

Table 9-6 ▪ Visual Studio Project Import Errors and Warnings (cont.)

Error or Warning Number	Description	Troubleshooting Tips
-9088	InstallShield does not have support for converting Visual Studio Web Setup projects into InstallShield projects.	This error occurs if you attempt to import or convert a Visual Studio Web setup project in InstallShield, since InstallShield does not have support for this conversion. If you encounter this type of error, consider creating a new project in InstallShield and using the Internet Information Services view to manage an IIS Web site on a target system.
-9089	InstallShield does not have support for converting Visual Studio CAB projects into InstallShield projects.	This error occurs if you attempt to import or convert a Visual Studio CAB setup project in InstallShield, since InstallShield does not have support for this conversion.
-9090	The Visual Studio project type could not be identified. Therefore, the project was not converted.	This error occurs if you attempt to attempt to import or convert a Visual Studio project type that InstallShield could not identify.
-10000	Conversion canceled by user.	This error occurs if you cancel the conversion process before it finishes.

InstallShield Custom Action Reference

This section describes each of the built-in InstallShield custom actions that are added automatically to InstallShield projects to support different functionality.

_serial_verifyCA_isx

Decreases the value of the property `SERIALNUMVALRETRYLIMIT` by 1.

_serial_verifyCA_isx_helper

Decreases the value of the property `SERIALNUMVALRETRYLIMIT` by 1.

CheckForProductUpdates

Uses FlexNet Connect to check for product updates.

This custom action launches an executable file called `Agent.exe`, and it passes the following:

```
/au[ProductCode] /EndOfInstall
```

CheckForProductUpdatesOnReboot

Uses FlexNet Connect to check for product updates on reboot.

This custom action launches an executable file called `Agent.exe`, and it passes the following:

```
/au[ProductCode] /EndOfInstall /Reboot
```

DLLWrapCleanup

Standard DLL wrapper that cleans extracted data.

This is a Windows Installer .dll custom action. The name of the .dll file is `dllwrap.dll`, and its entry point is DLLWrapCleanup.

DLLWrapStartup

Standard DLL wrapper that extracts data that describes calls.

This is a .dll custom action. The name of the .dll file is `dllwrap.dll`, and its entry point is DLLWrapStartup.

ISComponentServiceCosting

Extracts information from the `ISComPlusApplication` table and saves it in a temporary file for COM+ applications.

This is a Windows Installer .dll custom action. The name of the .dll file is `iscomsrv.dll`, and its entry point is ISComponentServiceCosting.

ISComponentServiceFinalize

Commits COM+ applications during installation and uninstallation.

This is a Windows Installer .dll custom action. The name of the .dll file is `iscomsrv.dll`, and its entry point is ISComponentServiceFinalize.

ISComponentServiceInstall

Installs COM+ applications during an installation.

This is a .dll custom action. The name of the .dll file is `iscomsrv.dll`, and its entry point is ISComponentServiceInstall.

ISComponentServiceRollback

Rolls back COM+ applications during a failed installation or uninstallation.

This is a .dll custom action. The name of the .dll file is `iscomsrv.dll`, and its entry point is ISComponentServiceRollback.

ISComponentServiceUninstall

Removes COM+ applications during uninstallation.

This is a .dll custom action. The name of the .dll file is `iscomsrv.dll`, and its entry point is ISComponentServiceUninstall.

ISIISCleanup

Removes temporary files and registry entries for an IIS installation.

This is a .dll custom action. The name of the .dll file is `IISHelper.dll`, and its entry point is ISIISCleanup.

ISIISCosting

Creates a list of actions in a temporary file for an IIS installation. Sets the `CustomActionData` property for other IIS actions.

This is a .dll custom action. The name of the .dll file is `IISHelper.dll`, and its entry point is ISIISCosting.

ISIISInstall

Creates Web sites, applications, virtual directories, and other items for an IIS installation.

This is a .dll custom action. The name of the .dll file is `IISHelper.dll`, and its entry point is ISIISInstall.

ISIISRollback

Removes Web sites, applications, virtual directories, and other items during rollback for IIS.

This is a .dll custom action. The name of the .dll file is `IISHelper.dll` and its entry point is ISIISRollback.

ISIISUninstall

Removes Web sites, applications, virtual directories, and other items during uninstallation for IIS.

This is a .dll custom action. The name of the .dll file is `IISHelper.dll`, and its entry point is ISIISUninstall.

ISInstallPrerequisites

Launches the installation of prerequisites that are associated with features.

This is a .dll custom action. The name of the .dll file is `PrqLaunch.dll`, and its entry point is InstallPrerequisites.

ISJITCompileActionAtInstall

Precompiles .NET assemblies at installation time.

This custom action launches a Microsoft executable file called `ngen.exe`.

ISJITCompileActionAtUnInstall

Deletes a precompiled .NET assemblies at uninstallation time.

This custom action launches a Microsoft executable file called `ngen.exe`.

ISLockPermissionsCost

Sets the `CustomActionData` property for the ISLockPermissionsInstall action.

This is a .dll custom action. The name of the .dll file is `ISLockPermissions.dll`, and its entry point is ISLockPermissionsCostAction.

ISLockPermissionsInstall

Sets permissions when the product is installed.

This is a .dll custom action. The name of the .dll file is `ISLockPermissions.dll`, and its entry point is ISLockPermissionsInstallAction.

ISNetApiInstall

Creates users and groups from an .ini file.

This is a Windows Installer .dll custom action. The name of the .dll file is `ISNetAPI.dll`, and its entry point is ISNetApiInstall.

ISNetApiRollback

Rolls back changes for users and groups.

This is a .dll custom action. The name of the .dll file is `ISNetAPI.dll`, and its entry point is ISNetApiRollback.

ISNetCreateIniForOneUser

Extracts operations to a temporary file for users and groups.

This is a .dll custom action called `ISNetAPI.dll`.

ISNetDeleteIniFile

Cleans up temporary files for users and groups.

This is a .dll custom action called `ISNetAPI.dll`.

ISNetGetGroups

Puts groups into a combo box.

This is a Windows Installer .dll custom action. The name of the .dll file is `ISNetAPI.dll`, and its entry point is ISNetGetGroups.

ISNetGetServers

Puts a list of servers into a combo box.

This is a Windows Installer .dll custom action. The name of the .dll file is `ISNetAPI.dll`, and its entry point is ISNetGetServers.

ISNetGetUsers

Puts a list of users into a combo box.

This is a Windows Installer .dll custom action. The name of the .dll file is `ISNetAPI.dll`, and its entry point is ISNetGetUsers.

ISNetSetLogonName

Stores the user, group, and server properties that were entered in the LogonInformation dialogs.

This is a Windows Installer .dll custom action. The name of the .dll file is `ISNetAPI.dll`, and its entry point is ISNetSetLogonName.

ISNetValidateLogonName

Verifies that a valid user name, server, and password combination were entered in the LogonInformation dialog

This is a Windows Installer .dll custom action. The name of the .dll file is `ISNetAPI.dll`, and its entry point is ISNetValidateLogonName.

ISNetValidateNewUserInformation

Stores the user, group, and server properties that were entered in the LogonInformation dialogs.

This is a Windows Installer .dll custom action. The name of the .dll file is `ISNetAPI.dll`, and its entry point is ISNetValidateNewUserInformation.

ISPrint

Prints the contents of a ScrollableText control on a dialog.

This is a Windows Installer .dll custom action. The name of the .dll file is `SetAllUsers.dll`, and its entry point is PrintScrollableText.

ISQuickPatchFinalize

Cleans shared reference counts for a QuickPatch.

This is a Windows Installer .dll custom action. The name of the .dll file is `QuickPatchHelper.dll`, and its entry point is ISQuickPatchFinalize.

ISQuickPatchFixShortcut

Reinstalls shortcuts for a QuickPatch.

This is a Windows Installer .dll custom action. The name of the .dll file is `QuickPatchHelper.dll`, and its entry point is ISQuickPatchFixShortcut.

ISQuickPatchHelper

Applies clean feature states for a QuickPatch.

This is a Windows Installer .dll custom action. The name of the .dll file is `QuickPatchHelper.dll`, and its entry point is ISQuickPatchHelper.

ISQuickPatchInit

Cleans component and feature states for a QuickPatch.

This is a Windows Installer .dll custom action. The name of the .dll file is `QuickPatchHelper.dll`, and its entry point is ISQuickPatchInit.

ISQuickPatchInit9X

Cleans component and feature states for a QuickPatch.

This is a Windows Installer .dll custom action. The name of the .dll file is `QuickPatchHelper.dll`, and its entry point is ISQuickPatchInit9X.

ISQuickPatchInit9X2

Cleans component and feature states for a QuickPatch.

This is a Windows Installer .dll custom action. The name of the .dll file is `QuickPatchHelper.dll`, and its entry point is ISQuickPatchInit9X2.

ISRunSetupTypeAddLocalEvent

Runs the AddLocal events that are associated with the Next button on the Setup Type dialog. This action needs to be called when the installation does not display the Setup Type dialog.

This is a Windows Installer .dll custom action. The name of the .dll file is `ISXExpHlp.dll`, and its entry point is RunSetupTypeAddLocalEvent.

ISSelfRegisterCosting

Extracts operations to a temporary file for self-registration.

This is a Windows Installer .dll custom action. The name of the .dll file is `isregsvr.dll`, and its entry point is ISSelfRegisterCosting.

ISSelfRegisterFiles

Registers self-registering files.

This is a Windows Installer .dll custom action. The name of the .dll file is `isregsvr.dll`, and its entry point is ISSelfRegisterFiles.

ISSelfRegisterFinalize

Cleans up temporary file from self-registration.

This is a Windows Installer .dll custom action. The name of the .dll file is `isregsvr.dll`, and its entry point is ISSelfRegisterFinalize.

ISSetAllUsers

Sets ALLUSERS per upgrade or initial installation requirements.

This is a .dll custom action. The name of the .dll file is SetAllUsers.dll, and its entry point is SetAllUsers.

ISSetTARGETDIR

Sets TARGETDIR to [INSTALLDIR].

This is a .dll custom action. The name of the .dll file is SetAllUsers.dll, and its entry point is SetTARGETDIR.

ISSetupFilesCleanup

Cleans up the temp directory of support files.

This is a .dll custom action. The name of the .dll file is SFHelper.dll, and its entry point is SFCleanupEx.

ISSetupFilesExtract

Extracts support files to a temp directory.

This is a .dll custom action. The name of the .dll file is SFHelper.dll, and its entry point is SFStartupEx.

ISUnSelfRegisterFiles

Unregisters self-registering files.

This is a .dll custom action. The name of the .dll file is issqlsrv.dll, and its entry point is ISUnSelfRegisterFiles.

LaunchProgramFileFromSetupCompleteSuccess

Launches an executable file at the end of an installation.

This is a Windows Installer .dll custom action. The name of the .dll file is SerialNumCAHelper.dll, and its entry point is LaunchProgram.

LaunchReadmeFileFromSetupCompleteSuccess

Launches a readme file at the end of an installation.

This is a Windows Installer .dll custom action. The name of the .dll file is SerialNumCAHelper.dll, and its entry point is LaunchReadMe.

setAllUsersProfile2K

Initializes the ALLUSERSPROFILE directory identifier.

SetARPINSTALLLOCATION

Resolves the directory identifier used in the Add or Remove Programs Read Me property. This custom action is required because ARPREADME is a Windows Installer property and such properties are not formatted automatically.

setUserProfileNT

Initializes the USERPROFILE directory indentifier.

ShowMsiLog

Displays the Windows Installer log file in Notepad if the end user selects the **Show the Windows Installer log** check box in the SetupCompleteSuccess, SetupCompleteError, or SetupInterrupted dialogs and then clicks Finish. This works only with Windows Installer 4.0 and later.

Command-Line Tools

In addition to its graphical user interface, InstallShield provides several command-line applications that you can use during build time, for instance, as part of a batch process, or during run time to customize the behavior of your installation. This section describes the command-line parameters for those applications.

- IsCmdBld.exe
- MsiExec.exe
- Setup.exe

IsCmdBld.exe

You can build a release from the command line using `ISCmdBld.exe` for Express projects.

Syntax

The following statement illustrates running `ISCmdBld.exe` to build the release **Othello Beta**:

```
ISCmdBld.exe -p "C:\InstallShield 2022 Projects\My Othello Project\Othello.ise" -r "SingleImage" -c COMP
```

The first parameter in the example above, starting with `-p`, is the path to the .ise file that you would like to build. Next, `-r Othello Beta` is the type of media. The parameter `-c COMP` specifies that you would like your package to be compressed into one file.

If a command-line build completes without any errors, InstallShield sets the environment variable ERRORLEVEL to 0. If an error occurs during a command-line build, ERRORLEVEL is set to 1. If ERRORLEVEL is set to any other value, this typically indicates an invalid parameter was passed to `ISCmdBld.exe`, and the specific cause of the error is displayed in the Command Prompt window in which `ISCmdBld.exe` was running.

Command-Line Parameters

`ISCmdBld.exe` supports the following command-line parameters.

Table 9-1 ▪ Command-Line Build Parameters

Parameter	Description
-b <build location>	The fully qualified path to the folder where you want the output folders and files to be placed. UNC paths are acceptable. The built installation's files will be located in the DiskImages\Disk1 subfolder of the location that you specify. Enclose long file names in quotation marks. This parameter is optional.

Table 9-1 ▪ Command-Line Build Parameters (cont.)

Parameter	Description
-c <compression option>	This parameter allows you to specify whether you want to have your release compressed into one file or remain uncompressed in multiple files. The valid arguments for this parameter are COMP and UNCOMP. To specify that your release be compressed into one file, use the COMP argument. If you do not want your release compressed, use the UNCOMP argument. This parameter is optional. If the release name already exists, the configuration will be based on what was specified in the IDE. If this is ignored for a new release, the new package will be uncompressed.
-customsign_args <arguments>	Use this parameter to specify the command-line argument for a custom signing tool's configuration. For example: `IsCmdBld.exe -p "C:\InstallShield 2026 Projects\MyProject1.ism" -sign_output "Y" -customsign_path "C:\Program Files (x86)\Windows Kits\10\bin\10.0.19041.0\x64\signtool.exe" -customsign_args "sign /v /a /fd SHA256 /tr http://timestamp.digicert.com /td SHA256 [filename]"` ***Important ▪*** *Specifying `-sign_output "Y"` is a prerequisite for using the `-customsign_path` and `-customsign_args` parameters.* The `[filename]` variable is a place holder for full file path to be signed. It resolves to full path of the binary file to be signed during build time. By default, a file path will be added at the end of an argument and passed to a custom sign tool. Instead of using a hard-coded path, you can utilize the path variables, environment variables, or property names that are defined within your project. ***Note ▪*** *When using a property name in place of the `[filename]` variable, ensure that it is enclosed within square brackets ([]).*
-customsign_path <path to signtool>	Use this parameter to specify the location of a custom signing tool that will be used to digitally sign the build-generated file (`setup.exe`). For example: `ISCmdBld.exe -customsign_path "C:\Users\Admin\Documents\sign.bat"`
-e <Y/N> *Setup Package*	Use this parameter to specify whether you want to create Setup.exe along with your installation project. Specify *Y* to build `Setup.exe`, or *N* to create just an installation package. This parameter is optional.
-fc <Launcher Copy Rights>	If you want to override the default copyright notice for `Setup.exe` with your required product copyright notice, use this parameter to specify the product copyright notice, such as, `-fc "Copyright © 2024 Flexera. All Rights Reserved"`.

Table 9-1 ▪ Command-Line Build Parameters (cont.)

Parameter	Description
-fd <file description>	If you want to override the default file description for `Setup.exe` with your required description, use this parameter to specify the file description, such as, `-fd "Setup Launcher Unicode"`.
-fv <version number>	If you want to override the default file version for **`Setup.exe`** with your own version number, use this parameter to specify the file version, such as `-fv 2.5.0.0`. This is an optional parameter. ***Note ▪*** *The file version always contains four fields. If you specify fewer than four fields for your file version, the remaining fields are filled with zeros. For example, if you specify a file version of 1.1, the file version that is used in the version resources of `Setup.exe` is 1.1.0.0.*
-i <.ini file path>	Instead of passing all of your parameters on the command line, you can include them in an initialization (.ini) file and call that file from the command line. For more information, see Passing Command-Line Build Parameters in an .ini File. Absolute and relative paths are acceptable. Enclose long file names in quotation marks. This parameter is optional.
-o <merge module search path>	This parameter specifies one or more comma-delimited folders that contain the merge module files (.msm) that are referenced by your project. InstallShield provides additional ways for specifying the folders that contain merge modules. For more information, see Specifying the Directories that Contain Merge Modules.
-p <project location>	Pass the path to the project (.ise) file. This path can be fully qualified, relative, or just the file name. UNC paths are also acceptable. If only the project file name is passed, the file is retrieved relative to the current working directory. For example: `IsCmdBld.exe -p "C:\InstallShield 2022 Projects\MyProject1\MyProject1.ise"` This is the only required parameter. Enclose long file names in quotation marks.
-prqpath <InstallShield prerequisite search path>	This parameter specifies one or more comma-delimited folders that contain the InstallShield prerequisite files (.prq) that are referenced by your project. InstallShield provides additional ways for specifying the folders that contain InstallShield prerequisite files. For more information, see Specifying the Directories that Contain InstallShield Prerequisites.

Table 9-1 ▪ Command-Line Build Parameters (cont.)

Parameter	Description
-r <release type>	Use this parameter to specify the release type (for example, SingleImage or WebDeployment). Valid options are: • Custom • CD_ROM • DVD-10 • DVD-18 • DVD-5 • DVD-9 • SingleImage • WebDeployment If you do not specify this parameter and build from the command line, the command-line build uses the last release type built in the IDE as the release type. If you have never built a release in the IDE, the command-line build uses the SingleImage release type. This parameter is optional.
-s	This parameter allows you to build your release in silent mode. Silent builds are useful if you want to run the build without displaying any errors or warning messages. This parameter is optional.
-se or -SE	This parameter allows you to run the build without displaying any information except error messages. Use this parameter to build and get only build errors (errors, fatal errors, and validation errors). This parameter is optional.
-sign_output <Y/N>	Use this parameter to specify whether the build-generated file (`setup.exe`) must be digitally signed. For example: • You can use `ISCmdBld.exe -sign_output "Y"` to enable digital signing. • You can use `ISCmdBld.exe -sign_output "N"` to skip digital signing.
-u	This parameter enables you to upgrade—but not build—your release. You can use this parameter to upgrade an installation project that you created with an earlier version of InstallShield. This parameter is optional.

Table 9-1 ▪ Command-Line Build Parameters (cont.)

Parameter	Description
-w	This parameter treats warnings that occur during the build process as errors. Each warning increments the error count by one. This parameter is optional.
-x	If you want the build to stop when it encounters an error, use the -x parameter. If you want the build to stop when it encounters a warning, use this parameter in conjunction with the `-w` parameter. This parameter is optional.
-y <product version>	This parameter enables you to specify a product version from the command line. This is especially helpful if you want to increment the build version (the third field) of the product version. For example, to set the product version to 1.0.5: `ISCmdBld.exe -y "1.0.5"` For information on valid product version numbers, see Specifying the Product Version. This parameter is optional.
cert_password	This parameter specifies the PFX digital certificate password or EV/ OV token password based on the signing options configured in the project. If it is not specified, the password configured in the project is used to sign the files. This is an optional parameter.

MsiExec.exe

`MsiExec.exe` is the executable program of the Windows Installer used to interpret installation packages and install products on target systems. After you build your release, you can install your Windows Installer package (.msi) from the command line. Currently, there is no support for passing Windows Installer parameters other than at the `Setup.exe` command line. To ensure that Windows Installer is installed on a target system, InstallShield creates a `Setup.exe` for your application by default. `Setup.exe` installs Windows Installer if it is not present on the target system or upgrades Windows Installer if an older version is present.

After building a release of your product, you can install it from the command line.

For an exhaustive `MSIExec.exe` reference, see Command-Line Options in the Windows Installer Help Library. For information on passing `MsiExec.exe` command-line parameters to `Setup.exe`, see Setup.exe.

Setup.exe

Like your compiled .msi file, Setup.exe can accept a limited number of command-line parameters. By using these parameters, you can specify which language you want your installation to run in and if you want `Setup.exe` to run silently. You can also pass parameters through `Setup.exe` to the included .msi file.

`Setup.exe` accepts the following command-line options, each of which is described below:

- /v : Pass command-line parameters to the .msi package
- /s : Silent
- /p : Password mode
- /a : Administrative installation
- /j : Advertise mode
- /x : Uninstall mode
- /f : Repair mode
- /ua : Install Windows 9x MSI Engine
- /uw : Install Windows NT MSI Engine
- /b : Cache installation locally
- /debuglog : Generate a log file for debugging
- /w : Wait
- /tempextractpath : Customize the Temp directory

Passing Parameters to the .msi File Within Setup.exe

If you include `Setup.exe` as part of your installation, you may need to pass command-line parameters to the .msi package stored within `Setup.exe`. To pass arguments to the .msi file, use the `/v` option. Once you specify this option, you can list any of the supported parameters that can be passed to Msiexec.exe. For example, to create a verbose log of the installation, enter the following:

```
Setup.exe /v"/l*v \"c:\My Log Files\test.log\""
```

There are a few special formatting rules that you need to follow when passing a parameter in this way. First, you need to place a backslash (\) in front of any quotation mark that resides within existing quotes. For example, the command line above contains the following: `v"/l*v \"c:\My Log Files\test.log\"`. Because the path to the log file is a long path, you need to use quotes. However, since you need to have quotes around the complete argument, the command-line statement fails if you do not use the backslash in front of all internal quotes.

Another formatting rule dictates that there cannot be a space between the command-line option (/v) and the arguments that you are passing, as illustrated in the example above.

In addition, if you are passing multiple parameters using the `/v` option, you need to separate them with a space—for example:

```
Setup.exe /v"/l*v \"c:\My Log Files\test.log\" /qn"
```

This command creates a log file and runs the installation silently.

As an alternative, you can use the /v option multiple times at the command line, once for each argument, as in the following example:

```
Setup.exe /v"/l*v \"c:\My Log Files\test.log\"" /v"/qn"
```

Running Setup.exe Silently

If you do not want the `Setup.exe` file to display a progress bar when it launches, you can use the `/s` command-line parameter. For example, if you enter the following command-line statement, `Setup.exe` launches, but the user interface is not displayed:

```
Setup.exe /s
```

If you want the .msi setup to run silently as well, you need to pass the `/qn` command-line parameter through `Setup.exe` using the `/v` parameter—for example:

```
Setup.exe /s /v/qn
```

Note ▪ *If your installation is password protected, you must also pass the /p parameter.*

Specifying a Password from the Command Line

You can specify a password for a password-protected setup by using the /p parameter. If you run a password-protected setup in silent mode, you must specify the password from the command line or the installation will fail.

To run a password-protected setup silently, enter the following statement at the command line:

Setup.exe /s /v/qn /p"*password*"

Specifying the Installation Mode from the Command Line

You can define what mode you want your installation to run in by using the `/a` (Administrative), `/j` (Advertise), `/x` (Uninstall), or `/f` (Repair) parameters. Each of these options is explained in detail below.

Administrative

When you run an installation in administrative mode, you can install an installation image to the network, which allows any one with access to that directory the ability to install that installation on their local machine with the privileges of the administrator who ran the administrative installation. No additional parameters are necessary for this option.

Advertise

Advertisement is a type of “just-in-time” installation in which features are installed when they are requested from the installer and not installed immediately during installation. When you launch MsiExec.exe with the `/j <package>` option, the features are advertised on the end user’s system, but not immediately installed. In most cases, the end user has the option to advertise features in the Custom Setup dialog. For information on the proper syntax for this parameter, see Command-Line Options in the Windows Installer Help Library.

Uninstall

If you want to uninstall your installation from the command line without displaying the maintenance dialogs, you need to use the `/x` option—for example:

```
Setup.exe /v/x
```

No additional parameters are required.

Repair

When you launch an installation in repair mode, it makes sure that all portable executable (.exe, .com, .ocx, .tlb) and help (.chm and .hlp) files are present and uncorrupted. If the installation detects that one of these files is missing or corrupt, it attempts to repair the file. To launch an installation in repair mode from the command line, use the `/f` option. For information on the proper syntax for this parameter, see MsiExec.exe.

Specifying the Windows Installer Engine Location from the Command Line

From the command line, you can specify the location of the Windows Installer engine to download for Web-deployable installations. To do so, use the following syntax.

Windows 9x MSI Engine

```
Setup.exe /ua"http://www.installshield.com/msiengine20/InstmsiA.exe"
```

Windows NT MSI Engine

```
Setup.exe /uw"http://www.installshield.com/msiengine20/InstmsiW.exe"
```

Specifying Cache Location

To enable advanced Windows Installer functionality, such as feature advertisement and application repair, you can copy the installation to a location on the target machine. To do so, use the following syntax, including a fully qualified path:

```
Setup.exe /b"C:\Storage\MyCachedPrograms\"
```

Note ▪ *The installation will be cached at a subdirectory of the location specified. This subdirectory is named after the Package Code GUID of the installation. If there is a problem unpacking to that location, the installation will ask for an alternate location.*

This option can only be used for SingleImage, Custom, and WebDeployment builds.

Generate a Log File for Debugging

The `/debuglog` parameter lets you generate a log file for `Setup.exe`.

To generate a log named `InstallShield.log` in the same directory as the `Setup.exe` file, pass just the command-line parameter. Note that this does not work if the `Setup.exe` file is in a read-only location. For example:

```
Setup.exe /debuglog
```

To specify the name and location of the log file, pass the path and name, as in the following example:

```
Setup.exe /debuglog"C:\PathToLog\setupexe.log"
```

Requiring Setup.exe to Wait

Use the `/w` command line option to have `Setup.exe` wait until `MsiExec.exe` is finished before `Setup.exe` exits. In addition, the `/w` command also returns any return codes generated by `MsiExec.exe`.

If you are using the `/w` option in a batch file, you may want to precede the entire `Setup.exe` command-line argument with **`start /WAIT`**. A properly formatted example of this usage is as follows:

```
start /WAIT setup.exe /w
```

Customize Temp Directory

The `/tempextractpath` parameter lets you to customize the %TEMP% directory—which by default contains all executable files—to the desired location. A typical command is as follows:

```
Setup.exe /tempextractpath"C:\IS_temp\"
```

Note ▪ *Using this parameter is recommended when an environment restriction, such as an antivirus, restricts an application from running executables from the %TEMP% directory.*

End-User Dialogs

This section serves as a reference for the end-user dialogs that are available in InstallShield. You can access most of the dialogs in the Dialogs view.

Global Dialog Settings for All End-User Dialogs

Global dialog settings enable you to make changes to common dialog settings across multiple dialogs in your project.

When you click the Dialogs explorer in the Dialogs view, InstallShield displays the following global dialog settings.

Table 9-1 ▪ Global Dialog Settings

Setting	Description
Global Dialog Image	Enter the path and file name of the image that you want to use as the global dialog image, or click the ellipsis button (...) to browse to the file. Every dialog in the project with a large, full-dialog image—such as Install Welcome—displays this file as its image. The image fills the entire dialog space and must be 499 pixels wide by 312 pixels high. InstallShield uses the value that you enter for this setting as the default value in the Bitmap Image setting for each dialog that has a full-dialog bitmap image. To override the image selection for an individual dialog, select that dialog in the Dialogs view, and then modify the path in the Bitmap Image setting. ***Note*** ▪ *InstallShield uses the image that you specify for the Global Dialog Image setting for dialogs that are not listed in the Dialogs view. One example is the dialog that is displayed when the installation is interrupted. Therefore, you should provide a global dialog image if you want to alter the visual presentation of the installation.* ***Caution*** ▪ *If you change the value of the Bitmap Image setting for a particular dialog and then you change the value of this Global Dialog Image setting, the value in the Bitmap Image setting is overwritten with the value in the Global Dialog Image setting.*

Table 9-1 ▪ Global Dialog Settings (cont.)

Setting	Description
Global Dialog Banner	Enter the path and file name of the image that you want to use as the global dialog banner, or click the ellipsis button (...) to browse to the file. Every dialog in the project that displays a banner image across the top of the dialog—such as License Agreement—displays this file as the banner. The image must be 499 pixels wide by 58 pixels high. InstallShield uses the value that you enter for this setting as the default value in the Banner Image setting for each dialog that has a banner image. To override the image selection for an individual dialog, select that dialog in the Dialogs view, and then modify the path in the Banner Image setting. ***Caution ▪*** *If you change the value of the Banner Image setting for a particular dialog and then you change the value of this Global Dialog Banner setting, the value in the Banner Image setting is overwritten with the value in the Global Dialog Banner setting.*
Global Dialog Theme	Dialog themes are predefined sets of images that give your end-user dialogs a unified and distinctive look. Select the dialog theme that you want to use for the dialogs in your project. For more information, see Dialog Themes.

Table 9-1 ▪ Global Dialog Settings (cont.)

Setting	Description
Show All Users Option	Specify whether you want to give end users the option of installing your product for all users or for only the current user. Available options are: • **No**—InstallShield does not include the option that enables end users to specify how they want to install the product. • **Yes (Only Windows 7 and Later)**—If the target system has Windows 7 or Windows Server 2008 R2, InstallShield adds buttons to the Ready to Install dialog. The buttons let end users specify how they want to install the product. If elevated privileges are required, the shield icon is included on the all-users button. If an end user selects the all-users button, the **ALLUSERS** property is set to 2, and the **MSIINSTALLPERUSER** property is set to 1. If an end user selects the per-user button, the **ALLUSERS** property is set to 1, and the **MSIINSTALLPERUSER** property is not set. • **Yes (All Systems)**—If the target system has Windows 7 or Windows Server 2008 R2, InstallShield adds buttons to the Ready to Install dialog. The buttons let end users specify how they want to install the product. If elevated privileges are required, the shield icon is included on the all-users button. If an end user selects the all-users button, the **ALLUSERS** property is set to 2, and the **MSIINSTALLPERUSER** property is set to 1. If an end user selects the per-user button, the **ALLUSERS** property is set to 1, and the **MSIINSTALLPERUSER** property is not set. If the target system has Windows Vista or earlier, or Windows Server 2008 or earlier, InstallShield adds radio buttons to the Customer Information dialog. The radio buttons let end users specify how they want to install the product. If an end user selects the all-users radio button and the end user has elevated privileges, the **ALLUSERS** property is set to 1. If an end user selects the per-user button and the end user has elevated privileges, the **ALLUSERS** property is set to an empty string (""). The default value is No. To learn more, see Per-User vs. Per-Machine Installations.

Splash Bitmap Dialog

The Splash Bitmap dialog is the first dialog that end users see when they launch your installation. Generally, this dialog contains an image—a .bmp or .jpg file—that contains the logo and branding for your company and product.

This dialog is optional, and it is not selected by default.

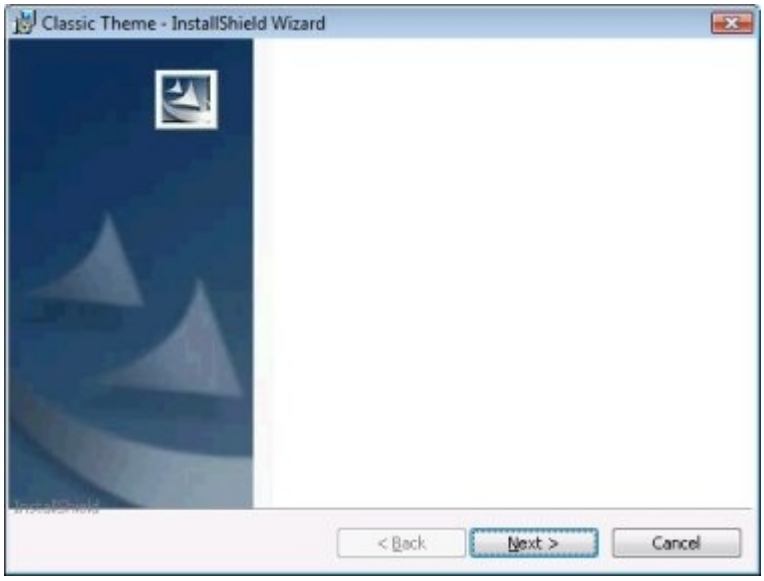


Figure 9-1: Splash Bitmap Dialog with Classic Theme

Figure 9-2: Splash Bitmap Dialog with InstallShield Blue Theme

Customizing the Dialog

You can customize this dialog by editing its settings. These settings appear to the right of the Dialogs explorer when you select the Splash Bitmap dialog. The Splash Bitmap dialog has the following settings:

Table 9-2 ▪ Splash Bitmap Dialog Settings

Setting	Description
Splash Bitmap	Enter the path and name of the image file (.bmp or .jpg) that you want to use for the splash screen dialog, or click the ellipsis button (...) to browse to the file. The image must be 465 pixels wide by 281 pixels high.
Sunken	If you want the splash image to appear sunken into the dialog, select Yes. If you want the image to appear flat in the dialog, select No.

Install Welcome Dialog

Depending on whether you choose to display the Splash Bitmap dialog, the Install Welcome dialog could be the first dialog displayed when your installation is launched. This dialog serves two purposes: The first is to let users know that your installation is running. To achieve this end, the name that you enter in the Product Name setting of the General Information view is displayed with a welcome message. The second purpose of the Welcome dialog is to display copyright information for your installation.

Although the Welcome dialog is required in every setup you create, you can customize its look and the information it provides.

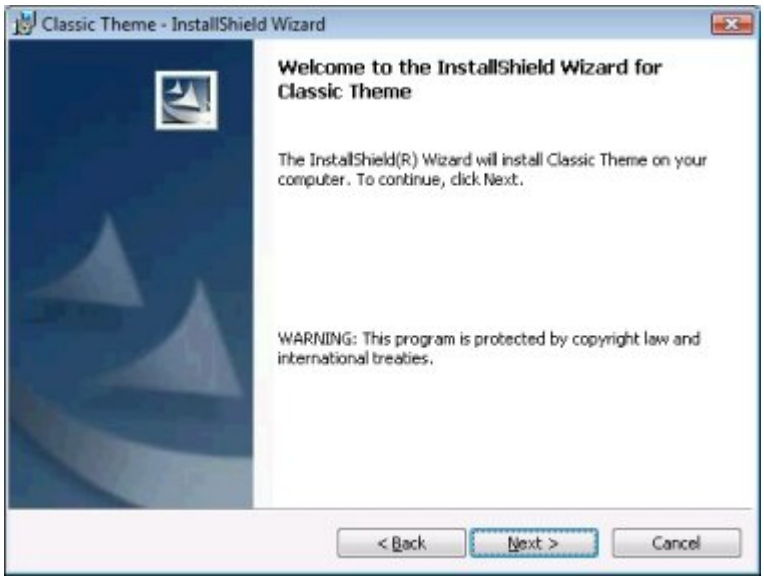


Figure 9-3: Install Welcome Dialog with Classic Theme

Figure 9-4: Install Welcome Dialog with InstallShield Blue Theme

Customizing the Dialog

You can customize this dialog by editing its settings. These settings appear to the right of the Dialogs explorer when you select the Install Welcome dialog. The Install Welcome dialog has the following settings:

Table 9-3 ▪ Install Welcome Dialog Settings

Setting	Description
Bitmap Image	Enter the path and name of the image file (.bmp or .jpg) that you want to use for this dialog's background, or click the ellipsis button (...) to browse to the file. The image must be 499 pixels wide by 312 pixels high.
Show Copyright	Specify whether you want to include copyright information for your product. If you select Yes, enter the appropriate text in the Copyright Text setting.
Copyright Text	Enter the copyright information for your product. Note that this information is not displayed unless you select Yes for the Show Copyright setting.

License Agreement Dialog

The License Agreement dialog displays your end-user license agreement (EULA). When this dialog is displayed, a user must accept your license agreement before continuing with the installation. Although this dialog is not required, it is selected by default.

Figure 9-5: License Agreement Dialog with Classic Theme

Figure 9-6: License Agreement Dialog with InstallShield Blue Theme

Customizing the Dialog

You can customize this dialog by editing its settings. These settings appear to the right of the Dialogs explorer when you select the License Agreement dialog. The License Agreement dialog has the following settings:

Table 9-4 ▪ License Agreement Dialog

Setting	Description
Banner Bitmap	Enter the path and name of the image file (.bmp or .jpg) that you want to use for this dialog's banner (across the top of the dialog), or click the ellipsis button (...) to browse to the file. The image must be 499 pixels wide by 58 pixels high.
License File	Enter the path to the rich text file (.rtf) that contains your license agreement, or click the ellipsis button (...) to browse to the file. ***Note ▪*** *The file must be in RTF format. Plain-text files do not work.*

Readme Dialog

The Readme dialog displays your application's readme file. This dialog is optional and is not selected by default.

Figure 9-7: Readme Dialog with Classic Theme

Figure 9-8: Readme Dialog with InstallShield Blue Theme

Customizing the Dialog

You can customize this dialog by editing its settings. These settings appear to the right of the Dialogs explorer when you select the Readme dialog. The Readme dialog has the following settings:

Table 9-5 ▪ Readme Dialog Settings

Setting	Description
Banner Bitmap	Enter the path and name of the image file (.bmp or .jpg) that you want to use for this dialog's banner (across the top of the dialog), or click the ellipsis button (...) to browse to the file. The image must be 499 pixels wide by 58 pixels high.
Readme File	Enter the path to the rich text file (.rtf) that contains your readme information, or click the ellipsis button (...) to browse to the file. ***Note*** ▪ *The file must be in RTF format. Plain-text files do not work.*

Customer Information Dialog

The Customer Information dialog lets you gather information such as name and company from end users. Additionally, you can link to a DLL that verifies serial numbers, in order to prevent unauthorized use of your software. Although this dialog is not required, it is selected by default.

The Customer Information dialog can also give end users the option of installing your product for all users or for only the current user. For more information, see Per-User vs. Per-Machine Installations.

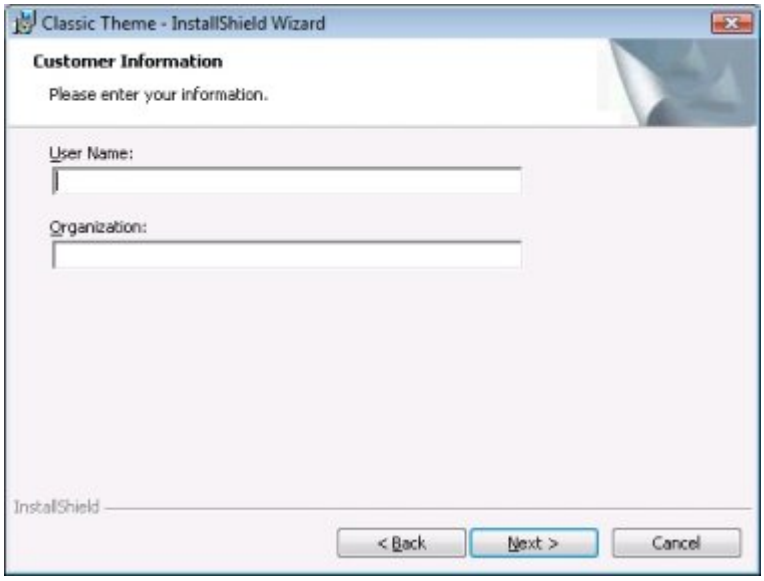


Figure 9-9: Customer Information Dialog with Classic Theme

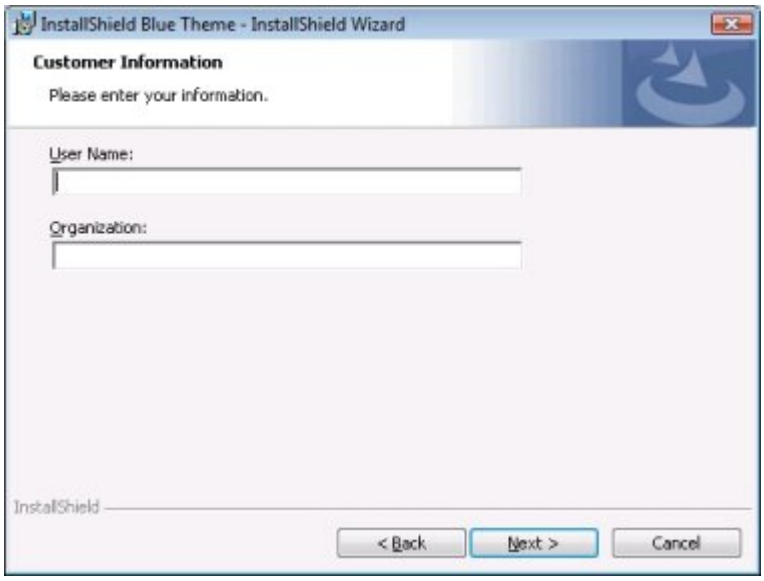


Figure 9-10: Customer Information Dialog with InstallShield Blue Theme

Customizing the Dialog

You can customize this dialog by editing its settings. These settings appear to the right of the Dialogs explorer when you select the Customer Information dialog from the explorer. The Customer Information dialog has the following settings:

Table 9-6 ▪ Customer Information Dialog Settings

Setting	Description
Banner Bitmap	Enter the path and name of the image file (.bmp or .jpg) that you want to use for this dialog's banner (across the top of the dialog), or click the ellipsis button (...) to browse to the file. The image must be 499 pixels wide by 58 pixels high.
Show Serial Number	Specify whether you want the dialog to have a serial number field. When an end user enters a serial number, the serial number can be verified by the DLL that you specify for the Serial Number Validation DLL setting. For more information, see Using a Custom Action for Serial Number Validation. If you select No, the other serial number settings are disabled.
Serial Number Template	Specify the format for your product's serial number: • Type a question mark (?) to represent each alphanumeric character. • Type a number sign (#) to represent each number. • Type a dash (-) between each group of characters. The dash indicates a break in the serial number, where one group of characters ends and another begins. For a serial number format of ###-????, the serial number field on the Customer Information dialog would consist of two boxes; end users would be able to type only three numbers in the first box and only four alphanumeric characters in the second box.
Serial Number Validation DLL	Enter the path to the DLL that contains the validation function for your serial numbers, or click the ellipsis button (...) to browse to this file. For information on how to format your function, see Using a Custom Action for Serial Number Validation.
Validate Function	Enter the name of the DLL function that validates your serial numbers. For example, if your DLL contains a function called **ValidateSN()**, enter the following: `ValidateSN` No special formatting is required. However, this function must exist in the DLL that you specified in the Serial Number Validation DLL setting.
Success Return Value	Specify the value that your validation function returns upon success. This return value's data type must be LONG. You can enter any number other than zero, since zero always signifies that the action failed.

Table 9-6 ▪ Customer Information Dialog Settings (cont.)

Setting	Description
Retry Limit	Specify the maximum number of times that a failed return value can be accepted before the installation exits. By entering a low number in this setting, you reduce the risk of someone guessing your serial number.

Destination Folder Dialog

The Destination Folder dialog displays the target destination for your installation and can provide functionality for changing that location. This dialog is optional and is not selected by default.

Figure 9-11: Destination Folder Dialog with Classic Theme

Figure 9-12: Destination Folder Dialog with InstallShield Blue Theme

Customizing the Dialog

You can customize this dialog by editing its settings. These settings appear to the right of the Dialogs explorer when you select the Destination Folder dialog. The Destination Folder dialog has the following settings:

Table 9-7 ▪ Destination Folder Dialog Settings

Setting	Description
Banner Bitmap	Enter the path and name of the image file (.bmp or .jpg) that you want to use for this dialog's banner (across the top of the dialog), or click the ellipsis button (...) to browse to the file. The image must be 499 pixels wide by 58 pixels high.
Show Change Destination	Specify whether you want this dialog to include the Change button, which enables end users to define a new destination folder for your product. Note that this button can also be displayed on the Setup Type and Custom Setup dialogs.

Database Folder Dialog

The Database Folder dialog displays the destination location of any database files that your installation installs. For example, if you create an installation for networked users and you want to install a common database for all users, you may want this database installed to the network rather than locally on every machine. You should use the DATABASEDIR setting in the General Information view to set the default destination folder for database files.

This dialog is optional and is not selected by default.

Figure 9-13: Database Folder Dialog with Classic Theme

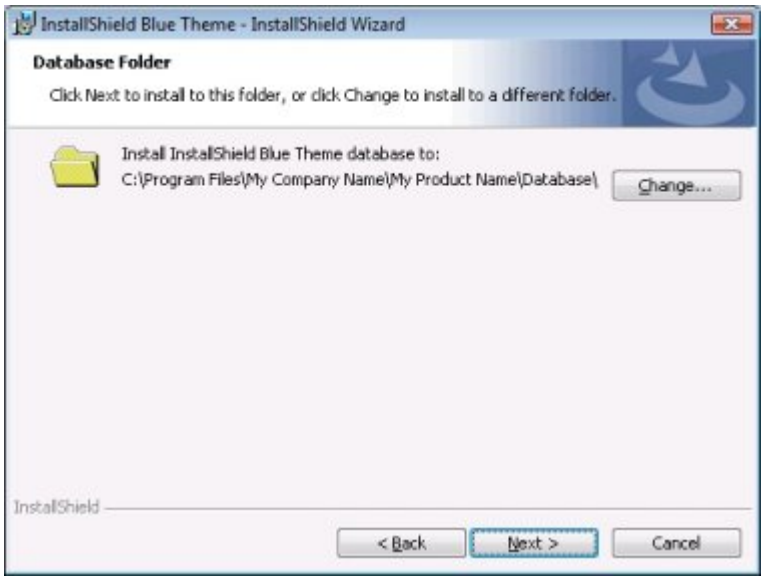


Figure 9-14: Database Folder Dialog with InstallShield Blue Theme

Customizing the Dialog

You can customize this dialog by editing its settings. These settings appear to the right of the Dialogs explorer when you select the Database Folder dialog. The Database Folder dialog has two settings:

Table 9-8 ▪ Database Folder Dialog Settings

Setting	Description
Banner Bitmap	Enter the path and name of the image file (.bmp or .jpg) that you want to use for this dialog's banner (across the top of the dialog), or click the ellipsis button (...) to browse to the file. The image must be 499 pixels wide by 58 pixels high.
Show Change Destination	Specify whether you want this dialog to include the Change button, which enables end users to define a new database destination folder for your product. Note that this button can also be displayed on the Setup Type and Custom Setup dialogs.

Setup Type Dialog

The Setup Type dialog displays the various setup configurations that you defined in the Setup Types view. It enables end users to select which setup type they prefer, thereby tailoring the installation to their needs. This dialog can also display the target destination for your product and provide functionality for changing that destination.

The Setup Type dialog is automatically included in your installation if you have more than one setup type selected in the Setup Types view. If you have only one setup type selected, this dialog is not displayed.

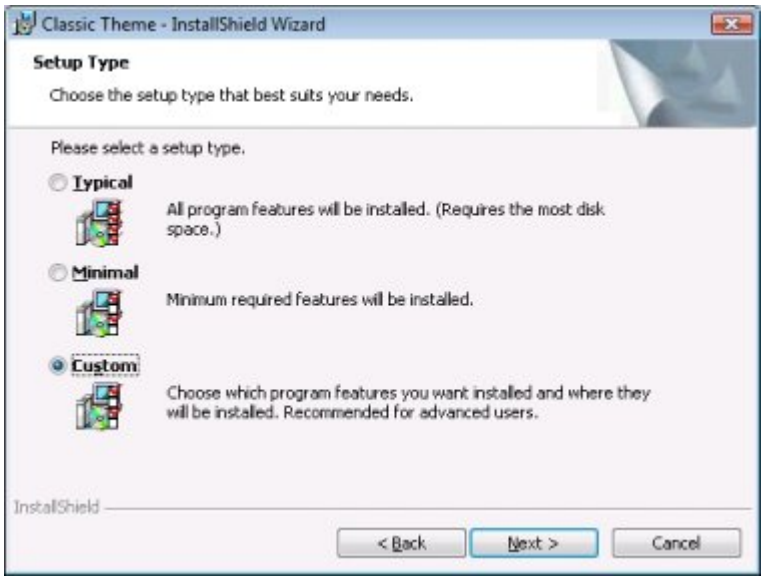


Figure 9-15: Setup Type Dialog with Classic Theme

Figure 9-16: Setup Type Dialog with InstallShield Blue Theme

Customizing the Dialog

You can customize this dialog by editing its settings. These settings appear to the right of the Dialogs explorer when you select the Setup Type dialog from the explorer. The Setup Type dialog has the following settings:

Table 9-9 ▪ Setup Type Dialog Settings

Setting	Description
Banner Bitmap	Enter the path and name of the image file (.bmp or .jpg) that you want to use for this dialog's banner (across the top of the dialog), or click the ellipsis button (...) to browse to the file. The image must be 499 pixels wide by 58 pixels high.

Custom Setup Dialog

The Custom Setup dialog is displayed only when a user selects the Custom Setup setup type. Therefore, in order to include this dialog in your setup, you must allow custom setups in the Setup Types view. In this dialog, end users can choose which features they want to install. This dialog can also display the target destination for your setup and provide functionality for changing that destination.

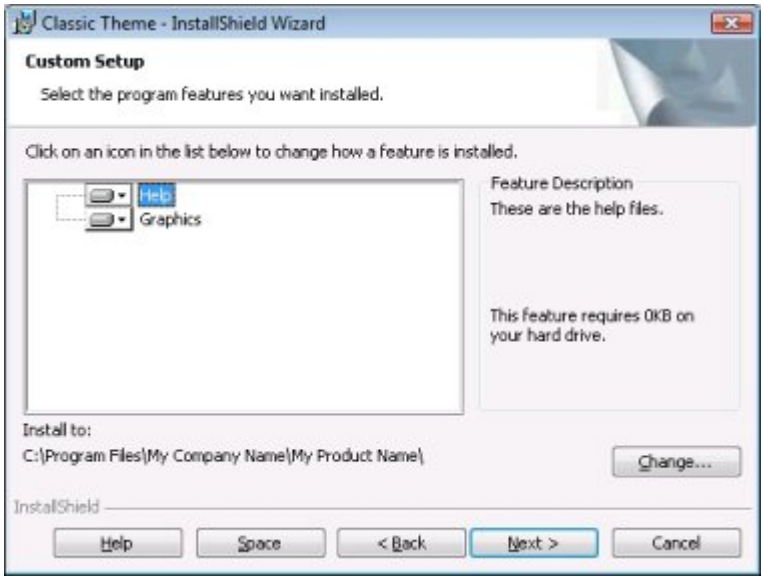


Figure 9-17: Custom Setup Dialog with Classic Theme

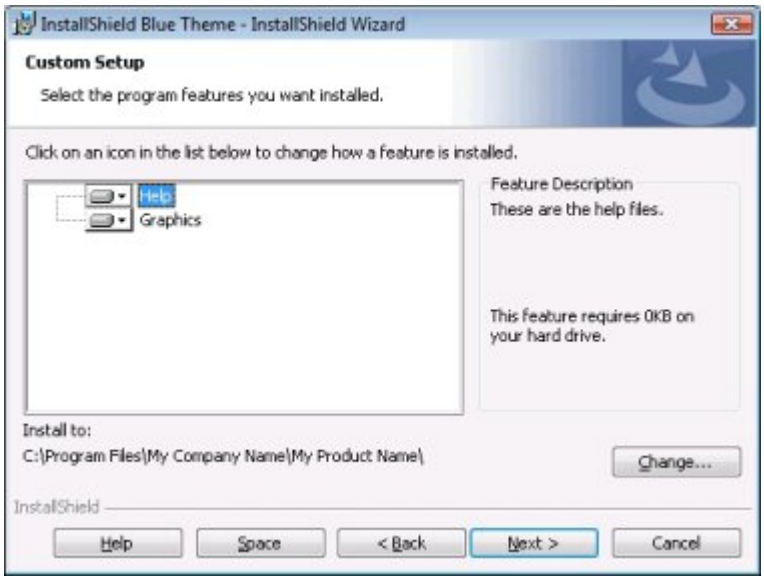


Figure 9-18: Custom Setup Dialog with InstallShield Blue Theme

Customizing the Dialog

You can customize this dialog by editing its settings. These settings appear to the right of the Dialogs explorer when you select the Custom Setup dialog. The Custom Setup dialog has the following settings:

Table 9-10 ▪ Custom Setup Dialog Settings

Setting	Description
Banner Bitmap	Enter the path and name of the image file (.bmp or .jpg) that you want to use for this dialog's banner (across the top of the dialog), or click the ellipsis button (...) to browse to the file. The image must be 499 pixels wide by 58 pixels high.
Show Change Destination	Specify whether you want this dialog to include the Change button, which enables end users to define a new destination folder for your product. Note that this button can also be displayed on the Destination Folder dialog.

Ready to Install Dialog

The Ready to Install dialog is the last dialog that occurs before file transfer takes place. Therefore, it is the end user's last chance to change installation options or cancel before the installation begins modifying the system. This dialog can also display a summary of the information that was communicated in earlier dialogs. Although this dialog is optional, it is selected by default.

The Install button on the Ready to Install dialog has the User Account Control (UAC) shield icon when the installation is run on Windows Vista or later systems and the installation is not yet running with elevated privileges.

The Ready to Install dialog can also give end users the option of installing your product for all users or for only the current user. For more information, see Per-User vs. Per-Machine Installations.

The Install button on the Ready to Install dialog has the User Account Control (UAC) shield icon when the installation is run on Windows Vista or later systems and the installation is not yet running with elevated privileges.

Tip ▪ *InstallShield is run with elevated privileges. Therefore, if you launch your installation from within InstallShield on a Windows Vista or later system, it has elevated privileges, and the UAC shield icon is not displayed on the Install button.*

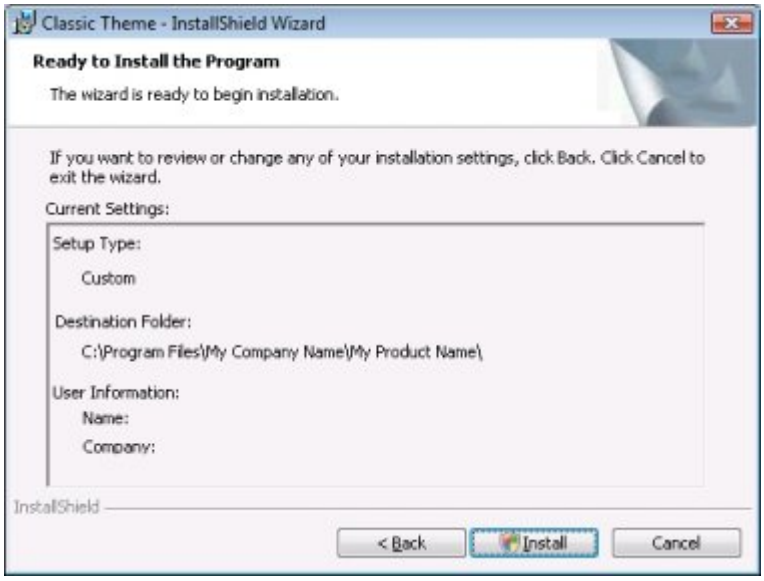


Figure 9-19: Ready to Install Dialog with Classic Theme

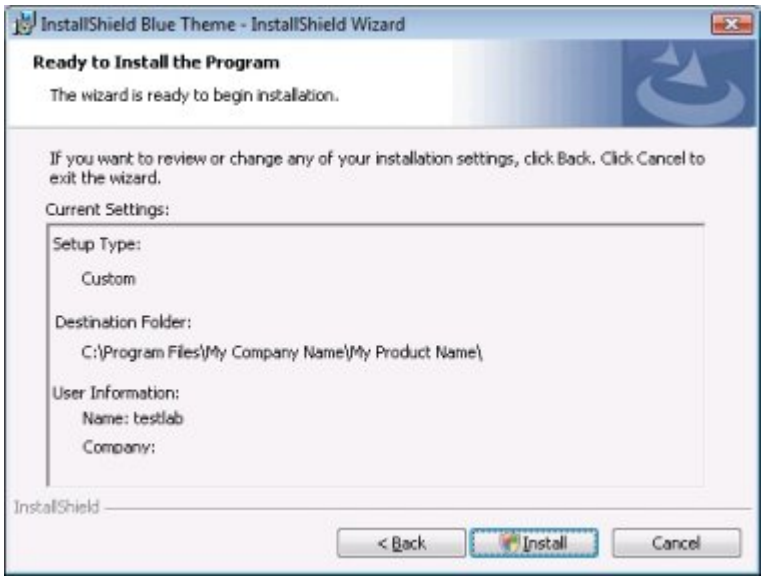


Figure 9-20: Ready to Install Dialog with InstallShield Blue Theme

Customizing the Dialog

You can customize this dialog by editing its settings. These settings appear to the right of the Dialogs explorer when you select the Ready to Install dialog. The Ready to Install dialog has the following settings:

Table 9-11 ▪ Ready to Install Dialog Settings

Setting	Description
Banner Bitmap	Enter the path and name of the image file (.bmp or .jpg) that you want to use for this dialog's banner (across the top of the dialog), or click the ellipsis button (...) to browse to the file. The image must be 499 pixels wide by 58 pixels high.
Show Current Settings	Specify whether you want this dialog to include a summary of the installation settings that the end user specified. Although end users cannot edit this information from the Ready to Install dialog, they can go back and change settings if necessary.

Setup Progress Dialog

The Setup Progress dialog enables you to display all the actions—such as file transfer—taking place during your installation. This dialog is not required.

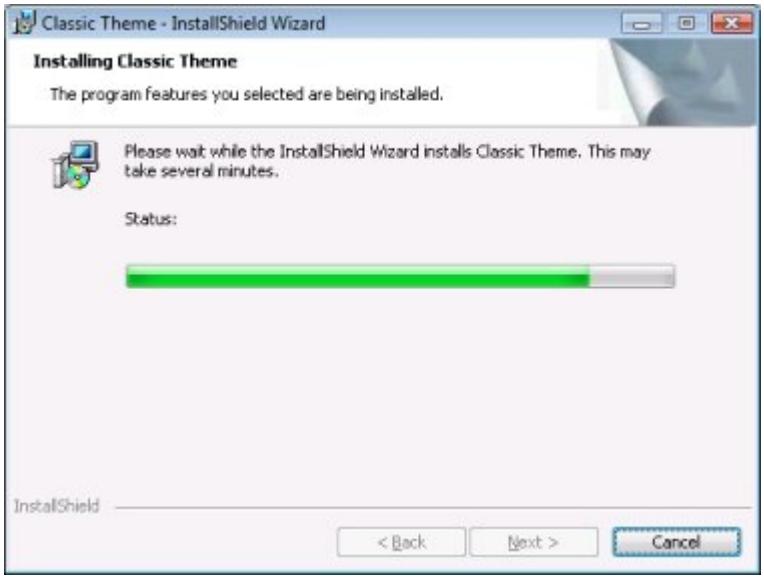


Figure 9-21: Setup Progress Dialog with Classic Theme

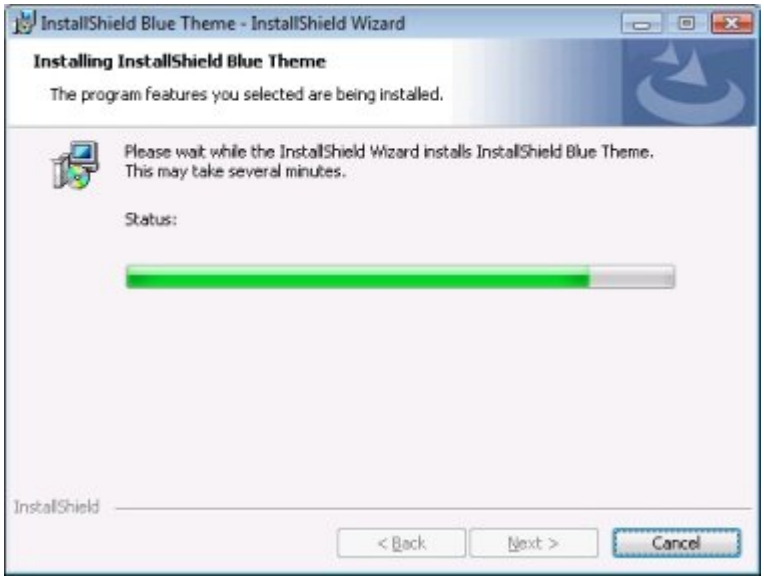


Figure 9-22: Setup Progress Dialog with InstallShield Blue Theme

Customizing the Dialog

You can customize this dialog by editing its settings. These settings appear to the right of the Dialogs explorer when you select the Setup Progress dialog. The Setup Progress dialog has the following settings:

Table 9-12 ▪ Setup Progress Dialog Settings

Setting	Description
Banner Bitmap	Enter the path and name of the image file (.bmp or .jpg) that you want to use for this dialog's banner (across the top of the dialog), or click the ellipsis button (...) to browse to the file. The image must be 499 pixels wide by 58 pixels high.
Show Progress Bar	Specify whether you want this dialog to include a progress bar. The progress bar gives end users a visual indication of how much of the total installation process has completed and how much is still remaining.

Setup Complete Success Dialog

When your installation successfully finishes, it displays the Setup Complete Success dialog. This dialog informs end users of a successful installation and optionally gives them the opportunity to check for updates for your product, launch your product, or view your readme file. This dialog is required.

If you enable Windows Installer logging through the General Information view, Windows Installer 4.0 creates a log file during installation of your product and populates the `MsiLogFileLocation` setting with the log file's path. In addition, a **Show the Windows Installer log check box** is added to the Setup Complete Success dialog. If the end user selects that check box and then clicks Finish, the log file is opened in a text file viewer or editor. To learn more, see Specifying Whether Windows Installer Installations Should Be Logged.

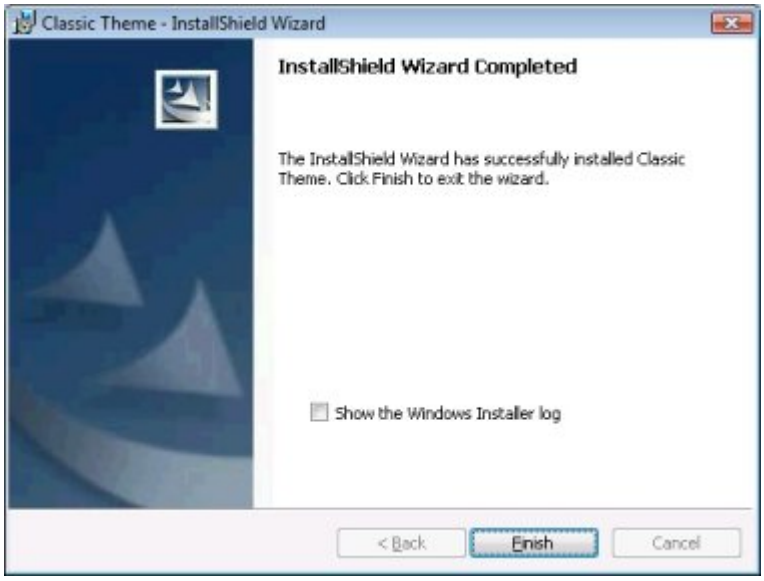


Figure 9-23: Setup Complete Success Dialog with Classic Theme

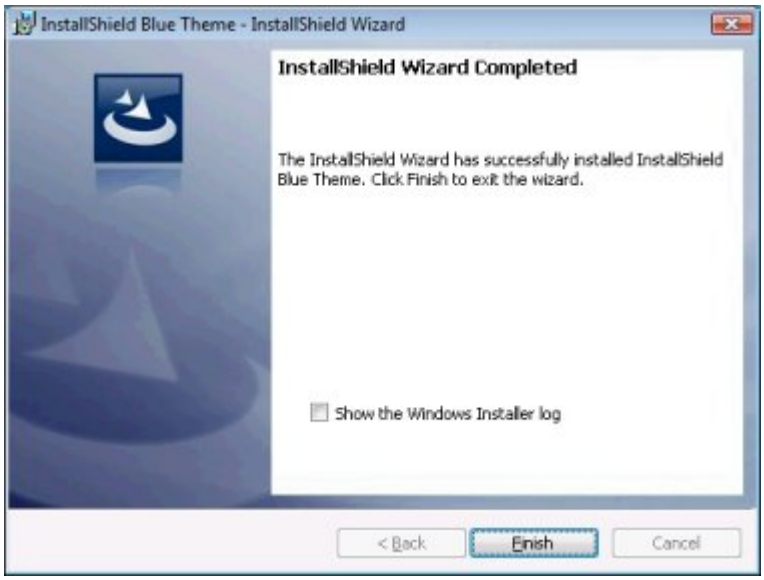


Figure 9-24: Setup Complete Success Dialog with InstallShield Blue Theme

Customizing the Dialog

You can customize this dialog by editing its settings. These settings appear to the right of the Dialogs explorer when you select the Setup Complete Success dialog. The Setup Complete Success dialog has the following settings:

Table 9-13 ▪ Setup Complete Success Dialog Settings

Setting	Description
Bitmap Image	Enter the path and name of the image file (.bmp or .jpg) that you want to use for this dialog's background, or click the ellipsis button (...) to browse to the file. The image fills the entire dialog space and must be 499 pixels wide by 312 pixels high.
Use FlexNet Connect User Interface	If you enable automatic update notification in your installation and you want to add a **Yes, check for program updates (Recommended) after the setup completes** check box to the Setup Complete Success dialog, select Yes for this setting. If an end user selects this check box and then clicks the Finish button to end the installation, FlexNet Connect is launched. For more information, see Notifying End Users about Upgrades Using FlexNet Connect. If you select Yes for this setting, the remaining settings for this dialog are not available. If you do not want the **Yes, check for program updates** check box included on the Setup Complete Success dialog, select No for this setting.
Show Launch Program	Specify whether you want the dialog to include a **Launch the program** check box. If the end user selects this check box and then clicks the Finish button to end the installation, your product is started once the installation exits. If you select Yes, you must use the Program File setting to select the executable file that you want to launch.
Program File	Select the executable file that you want to be launched if the end user selects the **Launch the program** check box. Note that Yes must be selected for the Show Launch Program setting.
Command Line Parameters	Enter any command-line parameters that you would like to pass to the executable file that you specified in the Program File setting.
Show Readme	Specify whether you want to give end users the opportunity to launch a readme file after the installation completes. If you select Yes, InstallShield adds a check box to the Setup Complete Success dialog that, when selected, causes the readme file that you specify in the Readme File setting to be launched.
Readme File	Enter the path to the file that you want to associate with the **Show the readme file** check box, or click the ellipsis (...) button to browse to this file.

MsiRMFilesInUse Dialog

Restarting the system after an installation is inconvenient for end users. One of the Windows logo program requirements is that all installations must contain an option that enables end users to automatically close applications and attempt to restart them after the installation is complete.

The MsiRMFilesInUse dialog is included in all Express installations by default. An installation displays the MsiRMFilesInUse dialog on a Windows Vista or later system if one or more files that need to be updated are currently in use during the installation. The dialog contains two options to allow end users to specify how to proceed:

- End users can choose to have the installation close the applications that are using those files and then attempt to restart the applications after the installation is complete.
- End users can avoid closing the applications. A reboot will be required at the end of the installation.

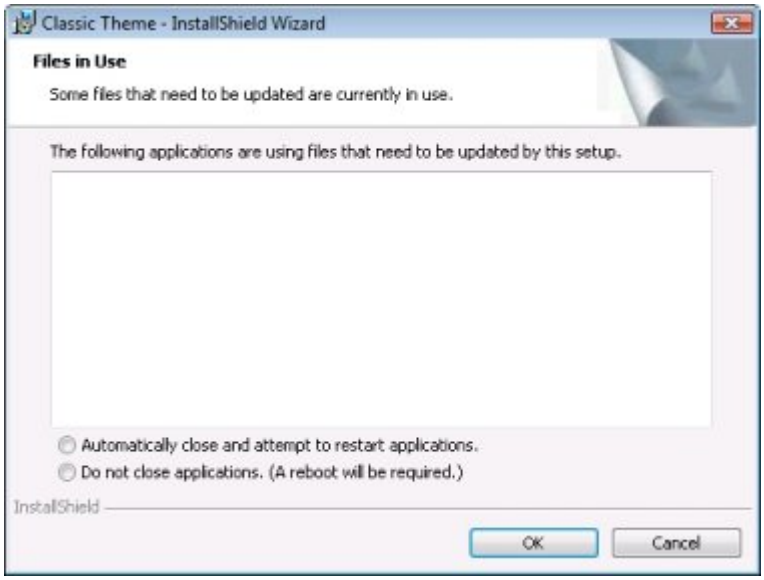


Figure 9-25: MsiRMFilesInUse Dialog with Classic Theme

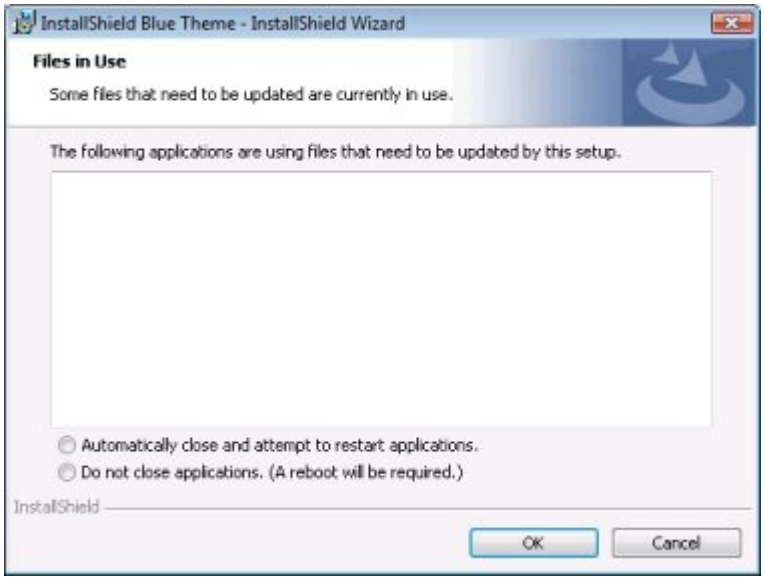


Figure 9-26: MsiRMFilesInUse Dialog with InstallShield Blue Theme

For more information about this dialog, see Minimizing Reboots on Windows Vista and Later Systems.

10

Frequently Asked Questions

This section lists common questions (sorted by topic) and provides links to answers.

- Where can I find information about Windows Installer run-time errors?
- How do I specify the destination folder for my application?
- How can I be sure the target machine has the Windows Installer service?
- How do I get changes I make in the InstallShield interface to take effect when I run my installation?
- How do I display a license agreement?
- How do I create shortcuts?
- How do I associate file types with my application?
- How do I launch an external application from my installation?
- How do I conditionally install my installation based on the operating system of the target machine?
- How do I upgrade my applications?
- How does Windows Installer determine which files should be overwritten?
- How do I create a self-extracting executable file for my installation?
- How do I change the product version during a build?
- How do I create a Web-based installation?
- Why will my installation not fit on a single floppy disk?
- How can I reduce the size of my installation?
- How can I create my installation so that it installs files based on the target operating system?
- What is the difference between extracting COM information and self-registration?
- What do I need to do to upgrade to the Premier or Professional Edition of InstallShield?

Answers to many other questions are available in the Knowledge Base.

Glossary

Table 1 ▪ InstallShield Express Glossary

Term	Definition
.bin file	A binary file that is executable on UNIX platforms. Application files usually have a `.bin` extension.
.bmp file	A bitmap file of an image represented as an array of bits. In bitmap graphics, an image is displayed on the screen as a collection of tiny squares called pixels, which together form a pattern. Each pixel in the image corresponds with one or more bits.
.cab file	See cabinet file.
.command file	A command file that is executable on Mac OS X platforms. Application files usually have a `.command` extension.
.cub file	A validation module that stores and provides access to custom actions for internal consistency evaluators (ICEs). See also custom action and internal consistency evaluator.
.exe file	An executable binary file on Windows platforms. Application files usually have an `.exe` file extension.
.gif file	A graphics interchange format file supported on the Web. This type of file is in a bit-mapped graphics file format used for displaying bitmap images. These files use lossless compression, a method of compression in which no data is lost. This type of file supports up to 256 colors; it also supports transparency, where the background color can be set to transparent to let the color on the underlying page show through. This type of file format is more suitable than the `.jpg` file format for images with only a few distinct colors, such as line drawings.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
.idt file	An exported Windows Installer database table.
.ism file	The working file that InstallShield uses to store project information. When you build a release, InstallShield uses the `.ism` file to create an `.msi` file for distribution.
.jar file	A Java archive file that contains the classes, images, and sound files for an installation, zipped up into a single file.
.jpg file	Also known as JPEG, which is the abbreviation for Joint Photographic Experts Group, the original name of the committee that wrote the standard. This type of file is one of the image file formats supported on the Web. The file uses lossy compression, a type of compression in which color and grayscale continuous-tone images are compressed and some data is lost to eliminate redundant or unnecessary information. These images support 16 million colors and are best suited for photographs and complex graphics.
.mif file	A management information format file used to describe a hardware or software component.
.msi file	The Windows Installer installation package in its finished state. It includes installation resource files and can have compressed within it all of the application's data files. The `.msi` file is the one that is distributed to the end users and will interact with the Windows Installer service to install your application on their machines.
.msm file	The database file of a merge module. This type of file contains all of the installation properties and installation logic for the module. See also merge module.
.msp file	A Windows Installer patch file. This type of file consists of a summary information stream, transform substorages, and cabinet files. An `.msp` file contains at least one database transform that adds patching information to the database of its target installation package. The installer uses this patching information to apply the patch files that are stored in the cabinet files.
.mst file	A transform package. This type of file is a simplified Windows Installer database that contains the differences between two `.msi` databases. Transforms enable administrators to apply modified settings to a database when they are deploying an installation package. See also transform.
.NET	An operating system platform that Microsoft created for connecting information, people, systems, and devices. The .NET environment enables developers to build, create, and deploy their applications and Web services using whatever languages they prefer via the common language run time.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
.NET Compact Framework	A scaled-down version of the .NET Framework designed to run on resource-constrained devices, such as mobile devices. The .NET Compact Framework is currently available only for Pocket PC and Windows CE .NET devices, with support for more devices forthcoming.
.NET Framework	A programming infrastructure that Microsoft created for building, deploying, and running applications and services that use .NET technologies such as Web services. The .NET Framework consists of three main parts: the common language run time, a hierarchical set of unified class libraries, and a componentized version of Active Server Pages called ASP.NET.
.ocx file	An object linking and embedding (OLE) custom control file. This type of file is a cross-platform COM file that is called by an application to perform a function, such as the ability to resize windows.
.p12 file	A certificate file that conforms to public-key cryptography standard (PKCS) number 12, which specifies a portable format for securely storing or transporting a user's private keys, certificates, miscellaneous secrets, and other information.
.pcp file	A Windows Installer patch creation properties file.
.prq file	A prerequisite file that contains information about a base application or component that must be installed on the target machine before the main application can be installed. Including setup prerequisites in installation projects enables developers to chain multiple installer files together into a single executable file.
.reg file	See registry file.
.scm file	A ScreenCam movie file.
.wmf file	A Windows metafile format file. This type of graphics file is used to exchange graphics information between Microsoft Windows-based applications.
absolute path	An absolute path includes all of the information necessary to locate a file by starting at the root directory of a specified drive. For example, `C:\Program Files\InstallShield` is the absolute path to the InstallShield folder when it is installed on drive C.
accessibility	The capacity or tendency to be available to all people, including those with disabilities.
acquisition phase	The phase of the installation process during which the installer queries the database for instructions. The acquisition phase is followed by the execution phase.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
action	A command that performs an operation at a particular point during the execution of an installation or uninstallation wizard. Developers of installation packages can use built-in standard actions and create their own custom actions. Actions may display progress to the end user or allow the end user to cancel the operation.
active directory	A structure supported by Microsoft Windows 2000 that enables administrators to track and locate any object on a network. Active Directory is the directory service used in Windows 2000 Server for distributed computing environments.
active template library (ATL)	A set of classes for writing COM controls, resulting in smaller binaries than, for example, MFC.
ActiveX	A set of object-oriented programming technologies and tools. ActiveX enables developers to write applications so that other applications and the operating system can call them. ActiveX technology makes it possible to create interactive Web pages that look and behave like applications, rather than static pages.
administration	The act or process of managing something.
administration sequence	The list of actions that are executed when a user launches your installation with the /a command-line option. This is useful for network administrators who want to provide a common installation point on the network and prepare different installation criteria for multiple users. Each Administration sequence consists of a User Interface sequence and an Execute sequence.
administrative installation	Copies and uncompresses your data files to a directory specified by the user, but does not create shortcuts, register COM servers, or create an uninstallation log.
administrative privileges	The highest level of permission that can be granted to a computer user. Levels of permissions are necessary in networked environments to ensure system security and prevent damage to computer hardware and software. A user with administrative privileges can perform tasks such as install and uninstall software and change a computer's configurations. Administrative privileges usually pertain to Windows NT 4.0, 2000, or XP machines as opposed to Windows 95, 98, or ME machines.
advertised shortcut	A shortcut to a product or feature that is not installed until the first time that the end user launches the shortcut. At run time of an installation, if the end user selects the "This feature will be installed when required" option for the product or the feature containing the shortcut, the shortcut is created but the component's files are not installed until the end user launches the shortcut. The first time the shortcut is launched, the Windows Installer service installs the component's files and other data and then the shortcut launches the target file. Every time the shortcut is used from then on, it behaves like a normal shortcut.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
advertisement	A type of "just-in-time" installation in which features are not installed immediately during the installation process. Instead, they are installed on the fly when they are requested. When you enable feature advertisement, the feature is advertised, regardless of the mode in which the installation is running, as long as no other factors prevent it from being advertised. If you launch `MsiExec.exe` from the command line with the `/jm` option, the feature is advertised to the end user's machine. If you use the `/ju` function, the feature is advertised to the current end user. In the Custom Setup dialog, the end user can control which features are immediately installed and which are available later. The two types of advertising are assigning and publishing. See also assigning; publishing.
advertisement sequence	The advertisement sequence contains the list of actions that are executed when a user launches your installation with the `/j` command-line option. Validation rule ICE78 requires the advertisement user interface sequence to be empty.
advertising	The process of presenting to end users features that are not installed immediately during the installation process. Instead, these features are installed on the fly when they are requested. See also advertisement.
AlwaysInstallElevated	A user policy that can be configured for Windows platforms under the following registry keys: HKEY_LOCAL_MACHINE\Software\Polices\Microsoft\Windows\Installer and HKEY_CURRENT_USER\Software\Polices\Microsoft\Windows\Installer. To install a package with elevated (system) privileges, set the `AlwaysInstallElevated` value under both of these registry keys to **`1`**.
American Standard Code for Information Interchange (ASCII)	The code that represents letters, numbers, punctuation marks, and other characters such as numbers, with each character assigned a value between 0 to 127. This code enables different types of computers and computer applications to exchange data.
animation	Multiple graphic images that are alternately changed and displayed at a specified number of frames per second to produce the illusion of movement.
API	See application programming interface.
app paths	The registry key that Windows uses to find your application and its `.dll` files if their locations are not already in the system's path. App paths entries can be set through the component's advanced settings.
application programming interface (API)	A set of routines that an application uses to communicate with the computer's operating system and that the operating system uses to make services available to the application.
ASCII	See American Standard Code for Information Interchange.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
assigning	A type of advertisement. If an administrator assigns an application to a machine, the installation program automatically runs the next time that the machine starts or restarts. If the administrator assigns an application to a user, the installation program places a shortcut on the Start menu of the user's machine. When the user selects the shortcut or launches a document associated with the assigned application, the application is installed.
asynchronous execution	A custom action that independently continues with the execution of its thread while the installer executes the main installation.
ATL	See active template library.
autonomic computing	A self-managed computing model named after, and patterned on, the human body's autonomic nervous system. An autonomic computing system would control the functioning of computer applications and systems without input from users, similar to the way that the human body's autonomic nervous system regulates and protects the body.
auto-repair	The automatic restoring of an application to its original state by the installer. A partial or complete application reinstallation might be required if any files associated with any feature are missing or corrupted.
basic UI	One level of the installer's internal user interface (UI) capabilities. Typically, installation packages that are built with the basic UI level display built-in modeless dialog that show progress messages and disc prompt messages. They do not display any authored dialogs.
billboards	Images, such as marketing messages, that can be displayed during installation.
Binary table	A table that holds the binary data for items such as bitmaps and icons. It also contains data for custom actions.
bitmap	A representation of an image presented as an array of bits. In bitmap graphics, an image is displayed on the screen as a collection of tiny squares called pixels, which together form a pattern. Each pixel in the image corresponds with one or more bits.
cabinet file	A single file that holds a number of compressed files. During installation of an application, the compressed files are decompressed and copied to your computer. Cabinet files are efficient because they save space and time when distributing software. A cabinet file usually has the file extension `.cab`. Missing or corrupt files may prevent installations from completing. It may be necessary to replace missing or corrupt operating system files or InstallShield files by extracting them from a cabinet file.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
cache	A cache (pronounced *cash*) is a temporary storage area for frequently accessed data. The purpose of caching is to store frequently used information in a location that is easy to access, resulting in a faster-running computer. There are two types of cache: memory cache and disc cache. Memory cache stores the data and the address of where the data is stored in main memory. Memory caching is useful because most applications access the same data repeatedly. Disc caching uses the main memory. It is used to hold information that has recently been requested from the hard disc or has previously been written to the hard disc. In general, installations usually use the disc cache. When data is read from or written to main memory, a copy is also saved in the memory cache. When data is called for, the computer first checks the memory cache, then the disc cache, and finally main memory.
caption	A text heading in a window. The windows caption is what users click and drag to position the active window on the screen.
CBT	See computer-based training.
CD	See compact disc.
CD browser	A graphical user interface that is launched when a CD is inserted in the drive. It is used to launch one or more applications on the CD.
CD-ROM	See compact disc read–only memory.
checksum	A calculated figure that is applied to data to test for possible corruption. The checksum is derived by sequentially combining bytes of data in the file in a systematic manner. After transmission or storage compression, the checksum calculation is performed again and the result is compared with the previous outcome. If the numbers do not match, this indicates that there is likely an error in the stored or transmitted data.
column	A vertical set of information.
COM	See component object model.
COM file	Binary software components containing reusable code that can be shared across products. A popular example of a COM file is a Microsoft Excel spreadsheet when it is embedded in a Word document. In this example, Excel acts as the COM server and Word acts as the client.
COM server	An executable file that exposes objects to other applications according to the Component Object Model (COM) specification. The types of objects exposed by a COM server may include ActiveX controls, ActiveX documents, Automation objects, or MTS components.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
COM+ file	An extended COM file. With regard to installations, COM+ adds greater support for building distributed components. For example, you can create appID keys and values in the registry and thereby configure various COM components to run remotely or with special privileges.
command	(1) An instruction given to a computer. (2) An instruction that the operating system executes from the command line or a Command Prompt window to perform a specific task. If errors occur during installation, you may, for example, need to execute commands to run utilities, search your computer's directories, or delete files.
command line	The area of the Command Prompt window in which commands are typed. A command line is used to pass commands to executable files (for example, `.exe`, `.bin`).
command prompt window	The Command Prompt window is the interface to MS-DOS. To access the Command Prompt window, click Run on the Start menu, and then type **`command.com`** in Windows 95, 98, and ME or **`cmd.exe`** in Windows NT, 2000, or XP. If errors occur during installation, you may need to open the Command Prompt window to enter commands to run utilities or search your computer's directories, for example.
command-line option	A command-line option is an argument to a command that changes how the command is executed. See also command.
commit execution	Execution of a Windows Installer action upon completion of the InstallFinalize action, which occurs when the installation has completed transferring files, registering COM servers, and creating shortcuts and registry entries.
committing databases	Accumulated changes made in a Windows Installer database. The changes are not reflected in the actual database until the database is committed, that is, until **MsiDatabaseCommit** is called.
compact disc (CD)	A disc used for electronically recording, storing, and playing back audio, video, and other information in digital form.
compact disc read-only memory (CD-ROM)	A type of compact disc that is only readable.
component	Elements of the application from the installation developer's perspective. Components are not visible to the end user. When the end user selects a feature for installation, the installer determines which components are associated with that feature, and then those components are installed. Components of an application would contain, for example, the executable binary files, data files, shortcuts, help system files, and registry entries.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
component object model (COM)	A Microsoft-developed software architecture that enables the creation of component-based applications. Component-based applications allow other components and other applications to use their features, adding functionality to these programs.
compress	Take one or more files and create a new file whose size is less than the total size of the original file(s) and from which the original file(s) can be re-created.
computer-based training (CBT)	Any instruction delivered by computer. CBTs often use graphics, sound, animation, and interaction to train people.
condition	A statement of logic that compares the value of a property to a fixed value or determines whether the property is defined. The condition must evaluate to true if the object, feature, component, action, or other item associated with the condition is to be installed or performed.
conditional installation	Installation of certain items only if certain conditions are met. For example, if an operating system condition is associated with the installation of an application, the application is installed on a target machine only if that target machine is running the specified operating system.
consume	What an installation does to a redistributable package such as a merge module or InstallShield object.
context menu	A context menu, also known as a right-click menu or a pop-up menu, opens when a user right-clicks an item on the desktop, in Windows Explorer, or in an application.
converted project	(1) A file that has been transformed from one type of file to another. (2) The Open MSI/MSM Wizard in InstallShield is a tool that converts `.msi` files and merge module (`.msm`) files to InstallShield installation projects (`.ism` files) that you can modify and build in InstallShield.
costing	File costing is the process that InstallShield uses to determine the total disc space that a current installation requires.
custom action	An action that is created by an installation author as opposed to a standard action that is built into a Windows installer. The action encapsulates a function performed during the installation, uninstallation, or maintenance of an application.
database	A discrete collection of data in a database management system (DBMS).
database function	A function that operates on a database.
database handle	A quantity that specifies a database when a database function is called.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
database management system (DBMS)	A set of applications that control the organization, storage, and retrieval of data for many users.
DBMS	See database management system.
DCOM	See distributed component object model.
DCOM file	Software components that are able to communicate across a network. DCOM, previously called network OLE, allows the components for a single application to be distributed across multiple networked computers.
decompress	The reverse process of compression, which minimizes the size of a file by removing its space characters. This process returns the file to its original state.
deferred execution	Execution of an installer action upon execution of the installation script.
dependency	A software object that is required by another software object.
dialog	Windows of information that display to the end user during installation and uninstallation. They enable the end user to interact with the operation by reading or specifying information. See also wizard.
dialog box	A window that displays within an application interface that enables users to enter information or specify commands.
differential release	A release that contains only those files that were absent from one or more of a specified set of existing releases. A differential release is used to update the versions of your product that were installed by those existing releases.
digital signing	To assure end users that the code within your application has not been tampered with or altered since publication, you can digitally sign your application. When you do so, end users are presented with a digital certificate when they download your product.
distributed component object model (DCOM)	An extension of the component object model (COM), a Microsoft-developed software architecture that enables the creation of component-based applications.
distribution media	A CD-ROM or other format of media deployed to users.
DLL	See dynamic link library.
drag	The act of selecting an item and moving it to another location.
dynamic link library (DLL)	A shared, code-base file containing functions that can be called from other applications.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
edit field object	An interactive object that lets the user enter text into a field.
elevated privileges	Privileges that are higher than standard privileges, and are usually temporarily. Privileges are permissions to perform certain actions on a system.
end user	The person who installs or uninstalls your product.
end user machine	The machine onto which an end user installs or from which uninstalls your product.
engine	A program that performs essential functions and coordinates the overall operation of other programs. Engines work behind the scenes. InstallShield uses the InstallScript engine (`ISScript.msi`) and the Windows Installer engine to drive installations. InstallShield also has its own proprietary engine called ikernel.
environment variable	A variable that can be accessed by multiple programs on the target system.
EXE media type	All of the installation and uninstallation files that are contained in a single, self-extracting file.
executable (.bin) file	A file that contains a program that can run on UNIX platforms.
executable (.exe) file	A file that contains a program that can run on Windows platforms.
execution phase	The phase during which an installer's actions are executed.
execution script	Installer actions for a Windows Installer installation. The execution script is generated during the acquisition phase of installation and executed during the execution phase.
expand	(1) To restore a compressed file to its original size. A compressed file is a single file that contains one or more other files, for example, a cabinet file. Compressed files are reduced in size, thus, saving space. (2) A DOS command used to restore compressed files. During an installation, missing or corrupted operating system files or InstallShield files may need to be expanded from a compressed file to enable the installation to complete.
extensible markup language (XML)	(1) A programming language that is essentially a simplified version of standard generalized markup language (SGML). It enables developers to create customized tags to organize and deliver content efficiently. (2) The format of the output of MultiPlatform projects. (3) Windows-based projects created in InstallShield can be converted to XML so that they can be saved in source code control software.
extensible stylesheet language (XSL)	A language used to describe how XML information should be presented.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
external user interface	The user interface developed by the author of an installation package. It does not use the internal user interface capabilities that are available with the installer.
extract	To remove, or decompress, a file from a compressed file, such as a cabinet file. A compressed file is a single file that contains one or more other files. Compressed files are reduced in size, thus, saving space. To use a file that has been compressed, it must first be "pulled out" of the compressed file. A command-line utility called `Extract.exe` can be used to extract files. During installation, missing or corrupted operating system files or InstallShield files may need to be extracted from a compressed file to enable the installation to complete.
false positive	A false positive is something that gives the appearance of being true by a test, but in reality is not. Most cases of false positives are found in anti-virus software. For example, anti-virus software may claim that it has found a virus in an InstallShield file, but in fact, this is not the case; their virus definitions need to be updated.
feature	Logical representations of the functionality of the product. For example, an installation could consist of a database, the main application files, and the help system files. Database, Application, and Help System would all be features of the product. Although features are visible to the end user, the actual files that comprise the features are within the components of the features. See also component.
file	An element of data storage in a file system.
file extension	The portion of a file name, following the final point, that indicates the kind of data stored in the file.
InstallShield scripting run time	See IDriver.exe.
formatted	When data has been divided or arranged for storage or display.
full update	An installation that installs an updated version of your product over an earlier version.
full user interface (UI)	One level of the installer's internal user interface (UI) capabilities. Installation packages that are built with the full UI level can display both the modal and modeless dialogs that have been included in the internal UI.
gallery	A collection of available resources that can be shared.
globally unique identifier (GUID)	A long string of numbers created by InstallShield to uniquely identify your product from others. Enter string GUIDs throughout InstallShield in the following format: {XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX}.
GUI	Graphical user interface.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
GUID	See globally unique identifier.
hard drive	The primary storage device on a computer. A hard drive contains disks on which data is read from and written to magnetically. The term “hard” differentiates the aluminum or glass disks used in a hard drive from floppy disks, which are made of plastic.
hotfix	A quick, important fix for a bug whose deployment cannot wait until the next release of the application. Most hotfixes are in the form of small patches and can be downloaded from the software vendor’s Web site.
HTML	See hypertext markup language.
hypertext markup language (HTML)	A language used to create Web pages with hyperlinks and markup for text formatting.
IAT	See import address table.
ICE	See Internal Consistency Evaluator.
IDE	See interactive development environment and integrated development environment.
IDriver.exe	The InstallShield scripting run-time engine. It is required on a computer to run some installations created with InstallShield. `IDriver.exe` is located in one or both of the following common file locations by default: `C:\Program Files\Common Files\InstallShield\Driver\7\Intel 32` or `C:\Program Files\Common Files\InstallShield\Driver\8\Intel 32`. Some installation error messages may reference `IDriver.exe`.
IE	See Internet Explorer.
ikernel.exe	The InstallShield engine. See also engine.
immediate execution	Installation phase terminology that generally refers to custom actions that are immediately executed when the installer builds the script in an InstallScript project.
import address table (IAT)	The table used to import `.dll` files or executable files.
indirect build	A full build that creates references to the payload files only. This means that at installation time, the payload must be in the same place where it was at build time. For example, if your payload is on a network server, then the installation looks for the payload on that same network server.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
install level	A feature's install level partly determines whether it is selected by default for installation. That value is compared to the Windows Installer property INSTALLLEVEL to determine which features are selected for installation: if a feature's install level property is less than or equal to the value of the INSTALLLEVEL, the feature will be installed.
installation	The transfer of a program and its constituent files, features, and components from source media to a target system. See also installation process.
installation package	One or more files that are used together to install or uninstall an application.
installation process	The entire process of installing the contents of a package or application onto a target computer. The installation process consists of the acquisition phase, the execution phase, and, if the installation attempt is not successful, rollback.
installation project	The entire collection of source files, dialogs, actions, and conditions that make up your installation and uninstallation while it is under construction.
installation sequence	The sequence in which an applications files are installed, acitons are executed, and dialogs display at run time. The installation sequence runs by default when the installation is launched, for example, by double-clicking the installation launcher.
installation type	A predefined group of features from which the end user can select to install or uninstall.
installation-on-demand	A Windows Installer capability that makes it possible to offer functionality to users and applications in the absence of the files themselves.
INSTALLDIR	A system variable that specifies the root destination directory for an installation.
installer	An installation and configuration service on a computer.
installer database	A database that contains all the necessary information for the installation of an application. It consists of many interrelated tables that together comprise a relational database of information necessary to install a group of applications.
installer function	The application programming interface called by an application to obtain installer services.
installer package authoring tool	A third-party tool that lets users create installation packages.
installer properties	Variables that are used during the installation.
INSTALLLEVEL	Initial level at which features are selected "ON" for installation by default.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
integrated development environment (IDE)	A software program's interface. Also referred to as an interactive development environment.
interactive development environment (IDE)	A software program's interface. Also referred to as an integrated development environment.
interface	The point at which independent systems communicate with each other. The user interface of an application often consists of toolbars, menus, buttons, windows, and other items.
Internal Consistency Evaluator (ICE)	Tests that can be run on a Windows Installer database (using Orca, Msival2, or the Premier or Professional editions of InstallShield) to warn about potential (or actual) authoring errors.
internal source files	Files that are included in the installation archive file.
internal user interface	Built-in capabilities of an installer that can be used to create a graphical user interface that is displayed to the end user during installation.
internet Explorer (IE)	The browser developed by Microsoft and distributed with their Windows operating systems.
ISM	See .ism file.
isolated application	An application that has been modified so that it always loads the versions of components, such as `.dll` files, with which it was originally developed and tested.
ISScript.msi	InstallScript engine installer. `ISScript.msi` installs the required files to run an installation. The InstallScript engine is also known as the InstallShield Scripting Run Time. InstallScript is a programming language used to create installations.
JAR	Acronym for Java archive file. See also .jar file.
Java Archive	See .jar file.
Java Native Interface (JNI)	A Java specification that enables Java code to call native code in a shared library. Native code in the JNI library can also invoke Java code through JNI.
Java Virtual Machine (JVM)	A simulated computer that interprets Java programs compiled into bytecode. The Java programs are usually stored in `.class` files.
JNI	See Java Native Interface.
JPEG	See .jpg file.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
JVM	See Java Virtual Machine.
KB	See knowledge base.
key file	A unique file for each component that Windows Installer uses to detect the component's presence. In order to create advanced component settings or shortcuts, a key file must be specified.
key path	A unique registry value for each component that Windows Installer uses to detect the component's presence. A component can have either a key file or a key path, but not both.
knowledge base (KB)	A collection of technical information, instructions, and articles that are beyond the scope of the software's help information. The Knowledge Base features tips, tricks, and techniques, answers to Frequently Asked Questions, and articles on both technical and design issues. Revenera posts periodic updates to the Knowledge Base Articles on its Web site (http://www.installshield.com).
left-click	The act of selecting the left button on a mouse.
library	A collection of similar objects that are stored for occasional use, such as programs in source code or object code form, data files, scripts, and templates. A program library is a collection of (usually) precompiled, reusable programming routines that a programmer can "call" when writing code.
load-ordering group	One service may require another service to already be running before it starts. For this reason, services need to be grouped and set to load in a specific order. Service load-ordering groups are listed under HKEY_LOCAL_MACHINE\System\CurrentControl Set\Control\ServiceGroupOrder. The service's Start Type property determines when it loads within its group.
localization	The process of adapting a product or service to a particular language and culture.
log database (LogDB)	A database that is on the target machine and contains a log of everything that was configured to be logged during installation and uninstallation.
log file	File containing a record of the activity that occurred during a software process (such as an installation, a build, a download) on a computer workstation or Web server. For example, Web servers maintain log files listing every request made to the server, summarizing files that were copied; bytes that were transferred; or which pages, images, and files are requested. A build log file lists all features, setup types, merge modules, dynamic links, and files included in the build.
LogDB	See log database.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
logging	The process of recording the activity of a software process (such as an installation, a build, or a download) in a log file.
macro	An instruction that represents a sequence of instructions.
maintenance mode	When a user runs an installation program a second (or later) time for a product already installed on their system, the installation runs in maintenance mode. Maintenance mode enables the user to modify feature selections from the first-time installation, repair the features already installed, or remove the entire program.
major upgrade	A comprehensive update of the product that warrants a change in the ProductCode property. A new product code is required if you want to have two versions of a product installed on the same machine.
managed application	An application is called a "managed application" if elevated (system) privileges are used to install the application.
MD	See media descriptor.
media descriptor (MD)	The semantics of an installable unit (IU). It provides a mapping of the elements in the deployment descriptor (DD) to some physical source for the data, which is usually files.
merge module (MM)	A package containing all of the logic and files needed to install distinct pieces of functionality such as run-time `.dll` files and virtual machines. Merge modules are built once and can be added to any installation project.
metafile	An image saved using the standard Windows metafile format. A metafile is a set of drawing instructions. Windows metafiles have a `.wmf` file extension.
MFC	See Microsoft Foundation Classes.
Microsoft Data Engine (MSDE)	A fully SQL Server–compatible data engine for building desktop and shared solutions. It provides an easy migration path to SQL Server 7.0. Solutions built with MSDE can be migrated to full SQL Server 7.0 without requiring a change in a single line of code.
Microsoft Developer Network (MSDN)	A set of online and offline services designed to help developers write applications using Microsoft products and technologies.
Microsoft disc operating system (MS-DOS)	The first operating system created by Microsoft. MS-DOS is the underlying operating system of Windows 95, 98, and ME. Windows NT, 2000, and XP operating systems support existing DOS applications.
Microsoft Foundation Classes (MFC)	A huge set of C++ classes that developers can use to write Win32 applications. To run an application that uses MFC, the MFC engine has to be installed on a target system.
MIDI	See musical instrument digital interface.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
migrating	The process of moving from the use of one operating environment to another that is, in most cases, thought to be a better one. For example, moving from Windows NT Server to Windows 2000 Server would be considered a migration because it involves ensuring that new features are exploited and that old settings do not require changing; it also involves taking steps to ensure that current applications continue to work in the new environment. Migration could also mean moving from Windows NT to a UNIX-based operating system (or the reverse). You can also migrate data from one kind of database to another kind of database. This usually requires converting the data into some common format that can be output from the old database and input into the new database. Migration is also used to refer simply to the process of moving data from one storage device to another.
MIME	See multipurpose internet mail extensions.
minor upgrade	An update to a product in which the changes made to the installation do not warrant a change in the Product Code.
MM	See merge module.
modal	Requires the user to interact with it before the application can continue, restricting the user's activities with other windows or dialog boxes. See also modeless.
modeless	Enables the user to interact with other windows and dialog boxes. See also modal.
MSDE	See Microsoft Data Engine.
MSDE named instance	A named instance of an installation of the Microsoft data engine (MSDE). See also named instance.
MSDE object template	A base for the MSDE merge module. When you add the MSDE merge module to a Windows Installer project or an InstallScript project, the InstallShield application starts with a base file (or stub) for this merge module. You need to configure this merge module, and then the application can apply your settings to the base file.
MSDN	See Microsoft Developer Network.
MS-DOS	See Microsoft disc operating system.
MS-DOS prompt	The Microsoft Disc-Operating System (MS-DOS) prompt is the visual indicator in the Command Prompt window signaling that MS-DOS is ready to accept a new command. The default MS-DOS prompt is `C:>`, followed by a blinking cursor.
MSI	See .msi file.
MSM	See .msm file.
MSP	See .msp file.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
MST	See .mst file.
multimedia	A collection of various media such as sound, video, graphics, and animation used together to convey a message.
multipurpose Internet mail extensions (MIME)	The standard used by Web servers to identify the files that they are sending to Web clients. The MIME standard is a way of specifying both the type of file being sent and the method that should be used to turn it back into its original form.
musical instrument digital interface (MIDI)	A protocol designed for recording and playing back music on digital synthesizers. Unlike `.wav` files, which are digital recordings of actual sound (voice or music), MIDI files simply define the instruments and notes that are to be played, and how they should be played. A synthesizer (generally part of a MIDI-capable sound board) on the end-user's system reads the MIDI instructions and plays the music. Since MIDI files contain only instructions and not the actual music, they are many times smaller than digital audio files of the same duration. MIDI files also use less processor power to play.
named instance	An instance is a complete and independent installation of SQL Server on a given server. One default instance and any number (up to 16) of named instances can be installed on a single server. Apart from the management tools and client connectivity components that are shared between instances, each instance is effectively standalone. Each instance has its own security, can be started and stopped independently, and can even be service packed independently of other instances on the same server.
nested installation	A type of custom action that installs or removes another installation package (sometimes called the child product) from within a running installation (called the parent product).
network	A network is two or more computers connected together to share hardware, software, and information. During installation, it may be necessary to move installation files that are on a network to your computer's hard drive to install the software properly.
NT service	An application that is installed on Windows NT to run as a service, meaning that once it is installed, execution is automatic and transparent to any user, and a user does not have to log in for the service to start. NT manages services that are defined and described in the registry.
object	A package containing all of the logic and files needed to install distinct pieces of functionality.
One Really Cool Application (ORCA)	An external tool for editing Windows Installer files.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
One-Click Install	With a One-Click Install installation, the end user can download an application with minimal effort and can begin using the application immediately. The installation is automatically downloaded, uncompressed, and then executed automatically, with minimal user input. End users are not asked to specify a download location, and they are not required to manually launch the installation.
operating system (OS)	The software that controls the operation of a computer and directs the processing of programs by assigning storage space in memory and controlling input and output functions.
ORCA	See One Really Cool Application.
OS	See operating system.
package	All of the files needed to run an installation, including the installation database file, your application's files (separate from the installation database file or compressed into it), and an executable file (which may have all of the above files compressed inside it).
package code	The globally unique identifier for an installation package. See also globally unique identifier (GUID).
panel	A window within a wizard that contains one or more related settings that an end user can configure. See also wizard.
parent/child installation	See nested installation.
patch	A special type of installation package that contains just the bits and portions of the application that is necessary to either update the application's files and installation to a specific version or to fix a bug in an earlier version.
patch file	A patch package used for patching. A patch package (`.msp`) file contains the transforms and instructions necessary for upgrading one or more installed versions of a product. Windows Installer uses a patch package to patch local or administrative installations. A patch package does not include a database like a regular installation package. Instead it contains at minimum one database transform that adds patching information to the database of its target installation package. The installer uses this information to apply the patch files that are stored in the cabinet file stream of the patch package.
patching	The method of updating an installation that replaces only the bits being changed rather than the entire application. This means that end users can download a patch for a product that is much smaller than the entire product.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
path	In a computer operating system, a path is the route through a file system to a particular file. A path name is the specification of that path. Each operating system has its own format for specifying a path name. The DOS, Windows, and OS/2 operating systems use the following format: `driveletter:directorynamesubdirectorynamefilename.suffix`. UNIX-based systems use the following format: `/directory/subdirectory/filename`.
path variable	A variable that represents a location that can be defined once in a central location so that it is not necessary to change every source file's path each time that the project is moved or the folder structure is changed. You can instead use path variables to define commonly used paths once, and they are used during the development of your installation project. These paths do not apply to the target machines where the application is being installed. Rather, they are used to link to source files that need to be included in your installation project. When the project is built, those links are evaluated and the files they point to are built into the installation package.
PKCS	See public-key cryptography standards.
platform	The operating system for which the installation program is intended. An installation technology may be limited to creating installations for a specific platform(s).
preview mode	Mode for viewing the design of the user interface, or the current appearance of dialog and billboards. *Preview mode* is a term used in Windows Installer.
product	The actual application to be installed or uninstalled.
product code	A string that uniquely identifies a product.
progress bar	The visual indication of the progress of an executable file.
property	A value that has been placed on an object within an installation project that is used during installation or uninstallation by the installer or uninstaller.
public property	A global variable whose value is set by the end user or system administrator. A public property can be set or changed during installation through interaction with the user interface as well as by setting the property on the command line, by applying a transform, or by using a standard or custom action. Unlike private property values, the values of public properties can be changed. Public property names cannot contain lowercase letters.
public-key cryptography standards (PKCS)	Developed by RSA Security, Inc., these specifications standardize aspects of public-key cryptography that are not covered by existing standards bodies.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
publishing	A type of advertising (“just-in-time” installation) in which no user-interface elements are created for the component during installation, but the component can still be installed through Add and Remove Programs of the Control Panel or when an installed component requests the published component from the installer.
qualified component	A method of single-level indirection that is similar to a pointer. Qualified components are primarily used to group components with parallel functionality into categories.
QuickPatch	A special type of patch package that contains just the bits and portions of the database necessary to update your application’s files and installation to a specific version. You can create a QuickPatch in InstallShield by using the QuickPatch type of project.
readme file	A text file that is included with the distribution of an application that contains important information. This information often pertains to the installation, uninstallation, or functionality that may not have been included in other product documentation.
red green blue (RGB)	The three colors of light that can be mixed to produce any other color. Colored images are often stored as a sequence of RGB triplets or as separate red, green, and blue overlays, though these are not the only possible representations. These three colors correspond to the three “guns” in a color cathode ray tube (CRT) and to the color receptors in the human eye.
redistributable	A merge module, object, or other file that may be legally distributed with an installation or application.
reduced user interface (UI)	The reduced UI level displays authored modeless dialogs, built-in modal error messages, and disc-prompt messages during installation. This UI level does not display any authored modal dialogs. This UI level uses the installer’s internal user interface capabilities. See also modal, modeless, and internal user interface.
reference count	A tally that is incremented each time a shared file is installed. It is maintained under HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\SharedDLLs.
refresh build	This type of build only recompiles custom code. It can also perform single file replacement, meaning it can only rebuild and add or modify a single changed file without rebuilding the entire installation. This type of build is available for InstallScript projects only.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
Regedt32.exe, Regedit.exe	The two versions of the Windows Registry Editor. The Registry Editor enables you to edit the entries in the registry. Regedt32 provides more functionality for editing the registry. If errors occur during installation, you may need to edit the registry to complete installation. `Regedt32.exe` and `Regedit.exe` have been merged in Windows XP computers, thus these two applications perform the same functionality in the new operating systems.
registry	A central database used by the Windows operating system to track the personal settings and the software and hardware installed on a computer. During installation, installation choices are written to the registry.
registry file	A text file of a predefined format that contains keys and values that can be merged into a registry.
reinstallation	When a product has been installed on a machine and its installation is run again, the installation reinstalls the product by overwriting its existing files, shortcuts, and registry entries.
relative path	A path that includes all of the information necessary to locate a file by starting at the current folder on the current drive, for example, `InstallShield\Support`. That folder can be located along that relative path only if it exists in the current directory.
release build	A full rebuild fit for releasing.
release notes	A file that is included with the distribution of an application. This file contains important information about the installation and uninstallation of the application. It can also contain information that may not have been included in other product documentation.
remote procedure call (RPC) stub	A small routine placed in a program that is used to request a service on another computer. RPC is a protocol, or agreed-upon format for transmitting data, that allows a program to request a service from a program located in another computer. The stub accepts the request from the program and forwards it to the remote procedure. When the procedure is complete, the stub receives the result and passes it back to the program that made the request.
repair mode	In repair mode, the installation checks for any type of corruption, such as missing or damaged files, incorrect registry entries, and self-registering files. If the installation detects any corruption, it attempts to repair the problem. See also maintenance mode.
resiliency	The ability for the application to reinstall components as necessary. If a component is accidentally deleted or corrupted, Windows Installer technology enables the application to essentially repair itself.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
restricted public property	A global variable for which the installation author can limit the ability to set or change it. Usually, only system administrators can manipulate restricted public properties. Restrictions are used to maintain a secure environment.
result set	The set of rows created by executing a **SELECT** statement.
RGB	See red green blue.
rich text format (RTF)	A Microsoft file format that contains special commands to indicate formatting information such as fonts and margins. RTF lets you exchange files between different word processors and operating systems.
rollback	The installation keeps track of all changes that are made during the installation process so that, if an error occurs and the installation is aborted, the changes will be "rolled back," that is, the machine will be restored to its original state.
row	A set of related columns that describe a specific entity, also known as a *record*.
RPC stub	See remote procedure call stub.
RTF	See rich text format.
run time	The time during which the installation interacts with the installer to install or uninstall your application on the target machine.
safe mode	Safe mode is a troubleshooting mode available in Windows 95, 98, ME, and 2000. When you start your computer in safe mode, only the operating system and mouse, keyboard, and display drivers are loaded. You may be able to start your computer in safe mode when it otherwise would not start at all. Safe mode lets you troubleshoot the operating system to determine what is not functioning properly.
schema	A description of the current structure of tables and views in a data source. The schema describes what columns are in each table, the data type of each column, and the relationships between tables.
SCM	See Service Control Manager.
SDK	See Software Development Kit.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
Section 508	A United States law that amended the Rehabilitation Act to require federal agencies to make their electronic and information technology accessible to people with disabilities. Section 508 was enacted to eliminate barriers in information technology, to make available new opportunities for people with disabilities, and to encourage development of technologies that will help achieve these goals. The law applies to all federal agencies when they develop, procure, maintain, or use electronic and information technology. Under Section 508, agencies must give disabled employees and members of the public access to information that is comparable to the access available to others.
security tool	A privacy tool that detects and eliminates destructive pests, such as Trojans, spyware, addware, and hacker tools on your computer. It complements anti-virus and firewall software, extending protection against non-viral malicious software that can evade the end user's existing security and invade privacy.
self-healing	See auto-repair.
self-registering file	A file that can enter information about itself in the Windows registry and remove that information upon uninstallation. Other types of files can be used without entering information into the registry. The installation of a self-registering file consists of installing the file to its desired location and then registering the file on your computer.
sequence	A collection of actions and dialogs that is executed sequentially in an installation project.
sequence tables	Tables that list the actions that control the installation process and specify their order of execution.
service	For Windows Installer–based or InstallScript-based projects, this is a program that runs in the background whenever a computer is running. Services perform tasks that do not require user interaction, such as software installation, process monitoring, file transfer, task scheduling, network management, and many more. In Win32, services are managed by the Service Control Manager.
Service Control Manager (SCM)	Maintains the system's database of services and exposes an interface for controlling these services.
service pack (SP)	An update to a software product that fixes existing problems and may provide product enhancements. The next version of a product incorporates all services packs previously released.
setup	See installation.
setup project	See installation project.
Setup.inx	The compiled script file. It is the object code that the installer engine executes.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
shortcut	A file that points to an application. Clicking the shortcut is a fast way to open the application. Shortcuts are usually placed on the desktop or on the Start menu. See also advertised shortcut.
silent installation	An installation that is run without a user interface or any end user intervention.
small update	A patch that upgrades a package where both the installed package and the most recent one have the same version number.
software development kit (SDK)	The documentation, samples, command-line compilers, debugging aids, utilities, and tools designed to enable developers to create applications and libraries that target a specific operating system. SDKs are usually provided by the manufacturer of the operating system.
source list	A list that specifies the locations where the installer searches for installation files. The entries in the source list can be network locations, URLs, or compact disks.
SP	See service pack.
splash screen	An image that is displayed to end users during the startup of a product installation. You can specify the image to be displayed, the length of time that it should display, and whether it contains localized images.
spyware	Any technology that aids in gathering information about a person or organization without their knowledge. On the Internet, spyware is programming that is put on someone's computer to secretly gather information about the user and relay it to advertisers or other interested parties. Spyware can be loaded on a computer as a software virus or as the result of installing a new program. Data collecting programs that are installed with the user's knowledge are not, properly speaking, spyware, if the user fully understands what data is being collected and with whom it is being shared.
SQL	See Structured Query Language.
SQL statement	A complete phrase in SQL that begins with a keyword and completely describes an action to be taken. For example, `SELECT * FROM Orders`.
standard action	An action that is built into an installation development software product. InstallShield products also support the creation and use of custom actions. See also action, custom action.
string table	A database that maintains the string IDs, values, and comments for all supported languages. These strings are used in dialogs and message boxes displayed to the end user at run time.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
Structured Query Language (SQL)	A language used to work with databases.
summary information stream	The properties such as title, author, package code, templates, summary, and schema that are defined for a Windows Installer package and are used by the installer to install the application.
synchronous execution	The opposite of asynchronous execution, where control of the process is not released until the entire process has completed.
system policy	The rules and regulations by which a system must abide.
system privileges	The system, programs, and functions available to each user.
table	A collection of rows of data.
TARGETDIR	In a Windows Installer–based installation, the TARGETDIR property specifies the root destination directory for the installation. In an InstallScript installation, the TARGETDIR system variable, by default, specifies the root destination directory for the installation.
Task Manager	A tool available on Windows platforms that provides information about the applications and processes running on your computer as well as your computer's performance. Prior to installation, you may need to use the Task Manager to end running applications and processes to prevent conflicts.
temp directory	A folder on your hard drive where the operating system or applications can temporarily store files while they are in use. When the application exits, the temporary files are deleted. It may be necessary to manually clean, or delete files from, the temp directory.
template	Something that establishes or serves as a pattern.
transform	A transform (`.mst`) file represents the differences between two installation databases. For example, network administrators may want to distribute different configurations of a product to the various departments in the company. As a result, you can create a transform for every configuration of the product, then apply the appropriate transform as needed.
transform error condition flags	A set of properties that are used to set the error conditions in a transform.
transform validation flags	The set of properties used to verify that a transform can be applied to the Windows Installer package.
tree node	A data structure that contains zero or more "child" nodes. Nodes are tree "roots". Tree nodes that do not have "children" are typically referred to as "leaves".

Table 1 ▪ InstallShield Express Glossary

Term	Definition
trees	Data that is structured like a tree.
UI	See user interface.
uncompress	See decompress.
uniform/universal resource locator (URL)	Represents the location of a Web site or page.
uninstallation	The undoing of an installation. Uninstallation is the installation maintenance option that enables the end user to remove the product files and reverse any changes that were made to the machine made during installation.
uninstallation log file	A record of all uninstallation-related events that occurred during an installation. The log file is initialized at the beginning of the installation. If this log file becomes corrupted, it may result in an error during uninstallation.
unInstallShield	A component included in the Setup Wizard that uninstalls the product from the end user's PC.
unspanned CD/DVD media type	The payload is laid out in a CD/DVD archive, but no spanning across multiple media is used.
Update Manager (Windows or Java)	An optional Win32 or Java-based client application that can be used by end users to set their own update schedules, or to silence the update queries performed by the agent.
upgrade	A move from a lesser version of an application to a newer, usually improved version of an application.
upgrade code	The code required to install a newer version of existing software.
upgrading	The process of installing a newer version of existing software.
URL	See uniform/universal resource locator.
user interface (UI)	The user's access to the software.
user profile	Record of an individual end user's settings, such as shortcuts, favorites, and settings for application, display, and hardware. User profiles enable multiple users to share a single computer while maintaining their own preferences.
validation	The process of validating, that is, to confirm the true form of data.
value pack (VP)	An installation that upgrades the features and functions of an installed application.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
variable	A value stored in the computer. Its value can be composed of any printable characters, numeric or text. Unlike a constant, whose value never changes, a variable's value can be changed at any time.
volume	The total amount of space, in blocks, on one piece of media.
VP	See value pack.
Web installation	An installation that end users can access and run from a Web site.
Web site	A location on the World Wide Web.
WinDir	Stores the path to the executable file for Win16 on Win32 (WOW).
window	The area in which an open application appears on a screen.
Windows API	The Windows application programming interface (API), which provides the building blocks that are used by applications written for the Windows platform. Each API is a specific method prescribed by a computer operating system or by an application program. A programmer writing an application can make requests of the operating system or another application with the use of a Windows API. Each API has different system requirements to run properly.
Windows installation CD	The compact disc that contains the Windows operating system. If your computer manufacturer installed the operating system on your computer, the installation CD should have been included with your computer documentation. If you installed the Windows operating system on your computer, you used the installation CD to do so.
Windows Installer	(1) An installation and configuration service. It is based on a data-driven model and provides all installation data and instructions in a single, complete package. In the data-driven installation model, a primary set of installation tables is created where every application resource (files, registry keys, and so on) is clearly tied to the component or feature it supports. The user selects the objects to install and where to install them, and the Windows Installer manages the procedural instructions. (2) Refers to the service, properties, and tables of an `.msi` package.
Windows Installer service	An operating system component that centrally manages application installation configuration and application uninstallation.
Windows NT service	See Windows service.
Windows service	Long-running executable applications that run in their own Windows sessions. For example, writing messages to an event log.

Table 1 ▪ InstallShield Express Glossary

Term	Definition
Windows system folder	Contains core operating system files, which are necessary to keep the computer running properly. Errors occurring during installation may be a result of missing or corrupt files contained in the System folder.
wizard	(1) A program utility that works as an interactive guide by walking the user step-by-step through an unfamiliar task. (2) The InstallShield Wizard is the wizard that displays during the run time of an installation or uninstallation. It presents the end user with a flexible, predefined series of dialogs that walk them through the installation or uninstallation process.
wizard dialog	Dialogs that display to the end user during installation and uninstallation. They enable end users to interact with the operation by reading or specifying information.
wizard interface	Used by the end user to interact with a wizard at run time.
WYSIWYG	What you see is what you get.
XCopyFile	An InstallScript function that copies one or more files and subdirectories to a target directory, creating subdirectories on the target machine, if necessary.
XML	See extensible markup language.
XSL	See extensible stylesheet language.

Index

Symbols

_IsSetupTypeMin 192
_serial_verifyCA_isx custom action 689
_serial_verifyCA_isx_helper custom action 689
.cab files
  configuring maximum size for 125, 126
  limitations 125, 126
.exe file 268
  custom actions 268
.ini file 531
  adding a section to 254
  creating a keyword 254
  pointing to a file 253
  view for 531
.isproj 326
.msi
  naming package file 577
.msi files
  running multiple ones simultaneously 214
.NET
  redistributables 227
.NET assembly 401
.NET Framework
  version 2.0, 64-bit 227
  version 3.0 227, 228
  version 3.0, 64-bit 227
.swf 308
.vdproj 150
  converting to an InstallShield project (.ise) 150

Numerics

27500 665
27501 666
27508 666
27509 666
27510 666
27517 667
27555 667
404 error messages for a Web site, application, or virtual directory 291
64-bit
  installing to 64-bit areas of the registry 241
  installing to 64-bit folders 193
  operating systems, targeting 85
  System32 folder source files 196

A

accessing the setup type at run time 192
actions, custom 565
  calling a DLL function 267
    setting properties for 263, 265
  calling an MSI DLL function 264
  JScript 269
  launching an .exe 268
    setting properties for 268
  scheduling in an installation or uninstallation 272
  VBScript 269
Add or Remove Programs in Control Panel 176
  information stored in 176
adding 401
  .NET assemblies to a project 401
  adding files to an IIS virtual directory 288
  InstallShield prerequisites to redistributables gallery 210
  InstallShield prerequisites, merge modules, and objects to a project 213
ALLUSERS 380

application lifecycle 98
assemblies 368
  patching in the global assembly cache 368
automatic update notification
  FlexNet Connect 510
autoplay 341

B

best practices
  dynamic file linking 201
billboards 305
  adding Flash files for 308
  adding images for 309
  configuring settings 309
  previewing without building and running a release 310
  purpose 305
  removing 311
  run-time behavior for 311
  screen shot samples 306
  setting the order of 310
  settings for Flash and image files 561
  specifying which type to use 308
  supported file types 305
  types 306
  view for adding and configuring 559
browsing for merge modules 211
building a release 315
  from the command line 323
  initiating a build 315
  installing Windows Installer 226
  performing quick builds 323
  preparing for release 576
  setting volume labels for multiple-disk installations 340
  single self-extracting installation file 332
  specifying the prerequisites location for 338
  troubleshooting build errors 619
  troubleshooting run-time errors 667

C

chaining installations 214
CheckForProductUpdates custom action 689
CheckForProductUpdatesOnReboot custom action 689
COM extraction
  excluding registry changes from 233
  with or without administrative privileges 88
COM server, registering 233
COM+ applications and components 553
  adding to your installation 279
command line 323
  building an installation from 323
  MsiExec.exe parameters for 701
  Setup.exe parameters for 701
command-line parameters for IsCmdBld.exe 697
CommonFiles64Folder 85
Company name
  displayed on Setup.exe's Properties dialog box 317
  displayed on Update.exe's Properties dialog box 366
components
  definition 96
compressed installation, creating 332
context-sensitive help 91
converting
  Visual Studio project to InstallShield project 150
copyright for Setup.exe 317
copyright for Update.exe 366
Create MSI Logs setting 494
creating
  compressed, self-extrating installation file 332
  InstallShield projects in Microsoft Visual Studio 398
  QuickPatch project 360
    for an existing QuickPatch 360
    overview 360
  setup projects 102
Custom actions
  _serial_verifyCA_isx 689
  _serial_verifyCA_isx_helper 689
  CheckForProductUpdates 689
  CheckForProductUpdatesOnReboot 689
  DLLWrapCleanup 689
  DLLWrapStartup 689
  InstallShield, descriptions of 689
  ISComponentServiceCosting 689
  ISComponentServiceFinalize 689
  ISComponentServiceInstall 690
  ISComponentServiceRollback 690
  ISComponentServiceUninstall 690
  ISJITCompileActionAtInstall 691
  ISJITCompileActionAtUnInstall 691
  ISNetApiInstall 691
  ISNetApiRollback 691
  ISNetCreateIniForOneUser 691
  ISNetDeleteIniFile 691
  ISNetGetGroups 691
  ISNetGetServers 692
  ISNetGetUsers 692
  ISNetSetLogonName 692
  ISNetValidateLogonName 692
  ISNetValidateNewUserInformation 692
  ISPrint 692
  ISQuickPatchFinalize 692
  ISQuickPatchFixShortcut 692
  ISQuickPatchHelper 693
  ISQuickPatchInit 693
  ISQuickPatchInit9X 693
  ISQuickPatchInit9X2 693
  ISRunSetupTypeAddLocalEvent 693

ISSelfRegisterCosting 693
ISSelfRegisterFiles 693
ISSelfRegisterFinalize 693
ISSetAllUsers 694
ISSetTARGETDIR 694
ISSetupFilesCleanup 694
ISSetupFilesExtract 694
ISUnSelfRegisterFiles 694
LaunchProgramFileFromSetupCompleteSuccess 694
LaunchReadmeFileFromSetupCompleteSuccess 694
setAllUsersProfile2K 694
SetARPINSTALLLOCATION 694
setUserProfileNT 694
ShowMsiLog 695
custom actions 565
calling a DLL function 267
setting properties for 263, 265
calling an MSI DLL function 264
including in QuickPatch 613
JScript 269
launching an .exe 268
setting properties for 268
scheduling in an installation or uninstallation 272
VBScript 269
custom error messages for a Web site, application, or virtual directory 291
custom images on dialog boxes 426
Custom Setup dialog 722
options for 302
sample 722

D

DATABASEDIR 488
dependencies 518
dynamic scanning 230
reviewing scanning results 230
static scanning 230
Dependency scanners
filtering files and 231
destination folder 488
General Information properties 488
specifying hard-coded destination directories 197
using registry key values as directory specifiers 178
Dialog theme 295
dialogs 558
digital signing
application 335
files 335
QuickPatch packages 364
timestamp server 123
DirectX 9 object 228
DirectX Object Wizard 460
Disk1 folder 276
DLL custom actions 265
settings 568
DLLWrapCleanup custom action 689
DLLWrapStartup custom action 689
downgrades, preventing 173
downloading redistributables to your computer 210
dynamic file linking 202
best practices 201
by-directory method 201
limitations 200

E

elevated privileges 169
End-user dialogs
themes for 295
Enhancements 30
Enhancements in InstallShield 2019 Express Edition 30
environment variables 535
properties 536
setting 257
view for 535
errors 619
build 619
Setup.exe run-time 667
errors, configuring for a Web site, application, or virtual directory 291
Executable
creating a single executable file for distribution
including InstallShield prerequisites when 219
executable
creating a single executable file for distribution
including InstallShield prerequisites when 214
Express project type 101

F

features 502
definition 96
making required 189
remote installation 190
visibility 188
file extension associations 533
creating 252
details 534
files 511
associating with features 515
finding in your project 204
including application files 193
linking dynamically 202
overwriting on the target machine 203
predefined destination folders 512
properties 430
scanning for dependencies 518

Filters.xml 231
specifying dependency scanner exclusions 231
specifying registry change exclusions for COM extraction 233
finding files and folders in your project 204
Flash file 308
adding as a billboard 308
FlexNet Code Aware integration 36
FlexNet Connect 510
view 510
Folder Properties dialog box 436
folders 204
finding in your project 204
forms authentication 293

G

global assembly cache 368
patching assemblies in 368
globalization 154
overview 154
supported languages 155
GUIDs 104

H

help
context-sensitive 91
using 91
HTTP errors, configuring 291

I

icons
for shortcuts 238
IIS 538
application mappings 290
ASP.NET version 289
configuring error messages for 291
creating a virtual directory 283
creating a Web site 283
creating an application 283
feature associations 289
host header 287
INSTALLSHIELD_SSI_PROP 282
nested virtual directory 284
overview 280
run-time requirements 281
site number 285
SSIEnableCmdDirective 282
SSL certificate 288
supported versions 280
TCP port number 285
timeout parameters 290
uninstalling applications 289
uninstalling virtual directories 289
uninstalling Web sites 289
view 538
IISROOTFOLDER support 293
images, custom, on dialog boxes 426
Install If Absent, Unnistall If Present option for registry keys 250
Install Only option for registry keys 250
install/uninstall behavior for registry keys 250
installation project types 101
installation projects 99
changing name or location 103
creating 102
opening 102
saving 103
typical elements 97
upgrading from an old version to a new version 128
working with 101
installations 96
definition 96
INSTALLDIR 376
InstallShield prerequisites 213, 214
adding to redistributables gallery 210
adding to your project 213
associating with features 216
configuring a release that includes 218
installation order for 217
overview 214
release location for 338
removing from features 217
removing from projects 214
removing from redistributables gallery 211
running installations with 220
setting a location for an individual one 219
setting the build-time location for 218
specifying required execution level 333
uninstalling applications with 223
User Account Control prompts and 169
InstallShield Prerequisites dialog box 439
INSTALLSHIELD_SSI_PROP property 282
Internet distribution 334
proxy server support 334
Internet Explorer prerequisite 214
Internet Information Services 538
application mappings 290
ASP.NET version 289
configuring error messages for 291
creating a virtual directory 283
creating a Web site 283
creating an application 283
feature associations 289
host header 287

INSTALLSHIELD_SSI_PROP 282
nested virtual directory 284
overview 280
run-time requirements 281
site number 285
SSIEnableCmdDirective 282
SSL certificate 288
supported versions 280
TCP port number 285
timeout parameters 290
uninstalling applications 289
uninstalling virtual directories 289
uninstalling Web sites 289
view 538
Invoker 333
IsCmdBld.exe 323, 697
ISComponentServiceCosting custom action 689
ISComponentServiceFinalize custom action 689
ISComponentServiceInstall custom action 690
ISComponentServiceRollback custom action 690
ISComponentServiceUninstall custom action 690
ISIISCleanup custom action 690
ISIISCosting custom action 690
ISIISInstall custom action 690
ISIISRollback custom action 690
ISIISUninstall custom action 690
ISInstallPrerequisites 690
ISJITCompileActionAtInstall custom action 691
ISJITCompileActionAtUnInstall custom action 691
ISLockPermissionsCost custom action 691
ISLockPermissionsInstall custom action 691
ISNetApiInstall custom action 691
ISNetApiRollback custom action 691
ISNetCreateIniForOneUser custom action 691
ISNetDeleteIniFile custom action 691
ISNetGetGroups custom action 691
ISNetGetServers custom action 692
ISNetGetUsers custom action 692
ISNetSetLogonName custom action 692
ISNetValidateLogonName custom action 692
ISNetValidateNewUserInformation custom action 692
ISO/IEC 19770-2 182
ISPreventDowngrade 141
adding or removing 173
ISPrint custom action 692
ISQuickPatchFinalize custom action 692
ISQuickPatchFixShortcut custom action 692
ISQuickPatchHelper custom action 693
ISQuickPatchInit custom action 693
ISQuickPatchInit9X custom action 693
ISQuickPatchInit9X2 custom action 693
ISRunSetupTypeAddLocalEvent custom action 693
ISSelfRegisterCosting custom action 693
ISSelfRegisterFiles custom action 693
ISSelfRegisterFinalize custom action 693
ISSetAllUsers custom action 694
ISSetTARGETDIR custom action 694
ISSetupFilesCleanup custom action 694
ISSetupFilesExtract custom action 694
ISUnSelfRegisterFiles custom action 694
Itanium
and .NET Framework 227

J

Java(TM) 2 Runtime Environment prerequisite 214
Jet 4.0 prerequisite 214
JRE prerequisite 214
JScript custom action 269

L

Lang ID 155
language identifiers 155
Launching
InstallShield on 32-bit vs. 64-bit systems 89
InstallShield with or without administrative privileges 88
LaunchProgramFileFromSetupCompleteSuccess custom action 694
LaunchReadmeFileFromSetupCompleteSuccesscustom action 694
lifecycle of an application 98
lock permissions for files and directories 450
lock permissions for the registry 452
locked-down environment 359
patches for 359

M

major Upgrade 345
managing COM+ applications and compontents 553
mapped-drive locations
referencing in projects 88
MDAC 2.8 prerequisite 214
merge modules 213
adding to your project 213
downloading to your computer 210
managing the gallery 212
removing from project 214
setting the build-time location for 224
Microsoft .NET Framework 227
version 2.0, 64-bit 227
version 3.0 228
version 3.0, 64-bit 227
minor Upgrade 346
MSBuild 326
MSDE 2000 prerequisite 214

MSI DLL custom actions 263
  settings 566
MsiPatchOldAssemblyFile 368
MsiPatchOldAssemblyName 368
MsiRMFilesInUse dialog 302

N

New Feature in InstallShield 2019 31
New Project dialog box 443
non-administrator patch 359
  requirements 359
notifying end users about updates
  FlexNet Connect 510

O

objects 213
  adding to your project 213
  downloading to your computer 210
  managing gallery of 208
  removing from project 214
ODBC resources 529
  associating with features 255
  attributes 256
  including in your installation 255
  installing 529
opening
  installation projects 102
Output window 414
  docking or undocking 122

P

package code 577
Palm OS device installations
  requirements 83
password protection
  for a QuickPatch 365
patch 357
  sequences for small updates 357
  uninstallation of 358
Patch Design view 347
patching
  assemblies in the global assembly cache 368
  project type 101
  vs. upgrading 349
  with non-administrator patches 359
  wizard 459
per-user vs. per-machine installations 258
  and HKEY_CURRENT_USER 251
Prerequisites
  associating with features 216
  removing from features 217
  setting a location for an individual one 219
  setting the build-time location for 218
  specifying required execution level 333
  User Account Control prompts and 169
prerequisites 213, 214
  adding to redistributables gallery 210
  adding to your project 213
  configuring a release that includes 218
  installation order for 217
  overview 214
  release location for 338
  removing from projects 214
  removing from redistributables gallery 211
  running installations with 220
  uninstalling applications with 223
Product name
  displayed on Setup.exe's Properties dialog box 317
  displayed on Update.exe's Properties dialog box 366
ProductCode 376
ProductName 376
products
  hierarchy of features and components for 96
ProductVersion 376
Program Compatibility Assistant 65
Program Files (x86) vs. Program Files 85
ProgramFiles64Folder 85
progress bar 306
  displaying with or without billboards 306
Project Assistant 105
project types 101
  converting Visual Studio project to InstallShield project 150
properties 376
  using 374
  Windows Installer Properties 376
proxy server settings 334

Q

quick builds 322
QuickPatch
  built-in InstallShield custom actions for 361
  how InstallShield creates 361
  project type 101
  streamlining creation of 361
  streamlining limitations 361
  wizard 459
QuickPatch projects 101
  custom action 613
  procedures for 360

R

Readme 488
reboot 273
redistributable 213
  adding to your project 213
  browsing for 211
Redistributables
  InstallShield 206
redistributables
  .NET Framework and language packs 227
  InstallShield prerequisites 214
  managing gallery of 208
registering COM servers 233
registry entries 243
  creating keys 243
  creating values 246
  drag-and-drop 244
  importing .reg files 465
  install/uninstall behavior for 250
  REG_EXPAND_SZ 248
  REG_MULTI_SZ 248
remote installation 190
removing
  InstallShield prerequisites from projects 214
  InstallShield prerequisites from redistributables gallery 211
  merge modules from projects 214
  objects from projects 214
Required Execution Level setting 333
requirements 564
requirements for target systems 83
Restarting target machine
  affect on UAC prompts for Windows Vista and later 169
running an installation
  running multiple .msi files simultaneously 214
  with InstallShield prerequisites 220

S

saving
  project with new name or location 103
searching for files and folders in your project 204
See cata source names (DSNs) 529
See end-user dialogs 558
See INI File Changes 531
See projects 99
Select Icon dialog box 454
self-extracting installation file, creating 332
sequences
  for installing prerequisites 217
  for small-update patches 357
serial number validation 272
services 257
  installing and starting 257
setAllUsersProfile2K custom action 694
SetARPINSTALLLOCATION custom action 694
setting uninstall behavior for registry keys 251
Settings.xml 123
setup types 505
  accessing at run time 192
  edit 191
  view for 505
Setup.exe 316, 332
  creating for your installation 332
  customizing properties for 317
  including InstallShield prerequisites in 214, 219
  installing Windows Installer with an application 226
  Setup.exe Command-Line Parameters 701
SETUPEXEDIR 376
setUserProfileNT custom action 694
Seup.exe
  renaming 577
shield icon 169
shortcuts
  icons for 238
shortcuts and program folders 520
  creating 237
    an application shortcut 237
  hot keys 239
  setting properties 521
  uninstallation 240
  view for 520
ShowMsiLog custom action 695
silent installation 342
small update 357
  patch sequences for 357
smart device installation
  project type 101
software identification tag 182
SourceDir 376
splash screen 709
SSL certificate 288
streamlining QuickPatch packages 361
subweb 283
  adding files to 288
  configuring error messages for 291
  creating 283
summary information stream 177
support files 576
Sysnative folder 196
system search 471
  file details 429
System64Folder 85
SysWow64 85

T

tagging 182
    to identify a product 182
target system requirements 83
TARGETDIR 376
Team Explorer 402
Team Foundation Server, integration with 402
text and messages 564
    editing in InstallShield 304
TFS, integration with 402
Theme, dialog 295
Tile configurations
    configurating desktop app tile on Start screen 241
    settings 527
timestamp server for digital signatures 123
toolbar
    adding InstallShield toolbars to the Visual Studio toolbar 399

U

UAC and installations 169
Uninstall Entire Key option for registry keys 250
uninstalling 223
    applications with prerequisites 223
    patches 358
Update.exe
    customizing properties for 366
upgrading 128, 505
    Express 2.x projects 143
    preventing downgrades while 173
    to a newer version of InstallShield 128
    vs. patching 349
    your product on a target machine 505
upgrading Express 2.x projects
    troubleshooting 663
User Account Control (UAC) and installations 169

V

VBScript custom action 269
views 487
    Configure the Target System 519
    Customize the Setup Appearance 558
    Define Setup Requirements and Actions 564
    list of, in InstallShield 487
    Organize Your Setup 487
    Prepare for Release 576
    Specify Application Data 510
virtual directory 283
    adding files to 288
    configuring error messages for 291
    creating 283
Visual Basic project type 101
Visual Studio 397
Visual Studio .NET
    adding InstallShield toolbars or commands to 399
    adding project output from Web Services or application 401
    integrating InstallShield with 397
    opening InstallShield projects in 398
Visual Studio Team Foundation Server
    integrating with InstallShield 402
voicewarmupx 440
VSSolutionFolder 399

W

Web deployment 335
Web services, deploying 292
Web site 283
    adding files to 288
    configuring error messages for 291
    creating 283
    forms authentication 293
Windows Installer engine requirements 83
Windows Installer redistributables
    version 4.5 227
Windows logo 173
Windows logo program 173
Windows Managment Instrumentation prerequisite 214
Windows Mobile device installations
    requirements 83
Windows services 257
    installing and starting 257
WMI prerequisite 214
Wow6342Node 85