Wise Package Studio

The software described in this book is furnished under a license agreement and may be used only in accordance with the terms of the agreement.

Documentation version 8.0

Legal Notice

Copyright © 2009 Symantec Corporation. All rights reserved.

Symantec, the Symantec Logo, and Altiris are trademarks or registered trademarks of Symantec Corporation or its affiliates in the U.S. and other countries. Other names may be trademarks of their respective owners.

The product described in this document is distributed under licenses restricting its use, copying, distribution, and decompilation/reverse engineering. No part of this document may be reproduced in any form by any means without prior written authorization of Symantec Corporation and its licensors, if any.

THE DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID. SYMANTEC CORPORATION SHALL NOT BE LIABLE FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS DOCUMENTATION. THE INFORMATION CONTAINED IN THIS DOCUMENTATION IS SUBJECT TO CHANGE WITHOUT NOTICE.

The Licensed Software and Documentation are deemed to be commercial computer software as defined in FAR 12.212 and subject to restricted rights as defined in FAR Section 52.227-19 "Commercial Computer Software - Restricted Rights" and DFARS 227.7202, "Rights in Commercial Computer Software or Commercial Computer Software Documentation", as applicable, and any successor regulations. Any use, modification, reproduction release, performance, display or disclosure of the Licensed Software and Documentation by the U.S. Government shall be solely in accordance with the terms of this Agreement.

Symantec Corporation
350 Ellis Street
Mountain View, CA 94043

http://www.symantec.com
Technical Support

Symantec Technical Support maintains support centers globally. Technical Support’s primary role is to respond to specific queries about product features and functionality. The Technical Support group also creates content for our online Knowledge Base. The Technical Support group works collaboratively with the other functional areas within Symantec to answer your questions in a timely fashion. For example, the Technical Support group works with Product Engineering and Symantec Security Response to provide alerting services and virus definition updates.

Symantec’s maintenance offerings include the following:

- A range of support options that give you the flexibility to select the right amount of service for any size organization
- Telephone and Web-based support that provides rapid response and up-to-the-minute information
- Upgrade assurance that delivers automatic software upgrade protection
- Advanced features, including Account Management Services

For information about Symantec’s Maintenance Programs, you can visit our Web site at the following URL:

www.symantec.com/techsupp/

Contacting Technical Support

Customers with a current maintenance agreement may access Technical Support information at the following URL:

www.symantec.com/techsupp/

Before contacting Technical Support, make sure you have satisfied the system requirements that are listed in your product documentation. Also, you should be at the computer on which the problem occurred, in case it is necessary to replicate the problem.

When you contact Technical Support, please have the following information available:

- Product release level
- Hardware information
- Available memory, disk space, and NIC information
- Operating system
- Version and patch level
- Network topology
- Router, gateway, and IP address information
- Problem description:
  - Error messages and log files
  - Troubleshooting that was performed before contacting Symantec
  - Recent software configuration changes and network changes
Licensing and registration
If your Symantec product requires registration or a license key, access our technical support Web page at the following URL:
www.symantec.com/techsupp/

Customer service
Customer service information is available at the following URL:
www.symantec.com/techsupp/
Customer Service is available to assist with the following types of issues:
- Questions regarding product licensing or serialization
- Product registration updates, such as address or name changes
- General product information (features, language availability, local dealers)
- Latest information about product updates and upgrades
- Information about upgrade assurance and maintenance contracts
- Information about the Symantec Buying Programs
- Advice about Symantec’s technical support options
- Nontechnical presales questions
- Issues that are related to CD-ROMs or manuals

Maintenance agreement resources
If you want to contact Symantec regarding an existing maintenance agreement, please contact the maintenance agreement administration team for your region as follows:

Asia-Pacific and Japan              customercare_apac@symantec.com
Europe, Middle-East, and Africa    semea@symantec.com
North America and Latin America    supportsolutions@symantec.com

Additional enterprise services
Symantec offers a comprehensive set of services that allow you to maximize your investment in Symantec products and to develop your knowledge, expertise, and global insight, which enable you to manage your business risks proactively.
Enterprise services that are available include the following:

Symantec Early Warning Solutions  These solutions provide early warning of cyber attacks, comprehensive threat analysis, and countermeasures to prevent attacks before they occur.
Managed Security Services         These services remove the burden of managing and monitoring security devices and events, ensuring rapid response to real threats.
Consulting Services  
Symantec Consulting Services provide on-site technical expertise from Symantec and its trusted partners. Symantec Consulting Services offer a variety of prepackaged and customizable options that include assessment, design, implementation, monitoring, and management capabilities. Each is focused on establishing and maintaining the integrity and availability of your IT resources.

Educational Services  
Educational Services provide a full array of technical training, security education, security certification, and awareness communication programs.

To access more information about Enterprise services, please visit our Web site at the following URL:

www.symantec.com

Select your country or language from the site index.
## Contents

### Technical Support

### Chapter 1: Introduction to Wise Package Studio

- About Wise Package Studio ................................................. 12
- Wise Package Studio Editions ........................................... 13
- Wise Package Studio Terminology ................................... 13
- Repackaging Basics ...................................................... 14
- Advantages of the Windows Installer Format ...................... 15

#### Starting Wise Package Studio

- If Your Logon Fails ..................................................... 17
- Wise Package Studio Logon Options ................................. 17

#### The Workbench Interface

- The Projects Tab ......................................................... 19
- When a Project Has No Process ..................................... 20
- When a Project Has a Process ....................................... 20
- The Tools Tab ............................................................. 21
- Resizing the Workbench Pane ....................................... 22

#### About the Wise Software Repository

- About the Share Point Directory .................................. 26
- How Source Files Are Indexed ...................................... 27
- Example: Populating the Share Point Subdirectories ............ 28

#### Integration with Software Virtualization Solution

- About Virtual Software Packages ................................ 30
- Wise Package Studio File Types .................................... 31
- Wise Package Studio Status Types ................................ 33
- Product Documentation .................................................. 33

### Chapter 2: Setting Up Wise Package Studio

- Steps for Setting Up Wise Package Studio ....................... 35
- Using the Initial Workbench Setup Project ....................... 38
- About Wise Package Studio Security .............................. 38
- Integrating With Windows NT Security ............................ 39
- Creating Groups and Setting Permissions ......................... 39
- predefined Security Groups ......................................... 41
- Creating Users ............................................................ 41
- Setting Software Manager and ConflictManager Security .... 43
- Setting SetupCapture Configuration Security .................... 44
- Setting Database Security ............................................. 46
- License Management .................................................... 47
- About User Licensing Setup ......................................... 48
- Adding Serial Numbers ............................................... 49
- About Evaluation Serial Numbers .................................. 50
- Assigning Licenses ...................................................... 50
- Deleting Serial Numbers .............................................. 51
- Workbench Preferences ................................................. 52
- Activating Suppressed Prompts ..................................... 53
- Setting Repository Preferences ..................................... 53
### Chapter 3: Creating Projects, Processes, and Tools ................................................................. 54
- About Projects ......................................................................................................................... 54
- Adding a New Project ............................................................................................................... 55
- Duplicating or Deleting a Project ......................................................................................... 57
- About Process Templates and Tasks ...................................................................................... 58
  - The Process Templates Setup Interface ........................................................................... 58
  - Predefined Process Templates ......................................................................................... 59
  - External Process Templates ............................................................................................. 60
- Adding a New Process ........................................................................................................... 61
- Adding Tasks to a Process ...................................................................................................... 62
- Duplicating and Deleting a Process ..................................................................................... 64
- Importing and Exporting Processes ..................................................................................... 65
- Organizing Tasks and Processes .......................................................................................... 66
- About Tool Setup ................................................................................................................... 66
  - Adding a New Tool .............................................................................................................. 67
  - Adding a Web Application as a Tool ................................................................................. 68
  - Duplicating, Deleting, and Rearranging Tools .................................................................. 69
- Help for Tasks and Tools ....................................................................................................... 70
  - Adding Wise Package Studio Variables to Help Text ...................................................... 71
- Command Line Options ....................................................................................................... 72
  - Guidelines for Entering Command Line Options ............................................................ 72
  - About Command Line Options for Tools ....................................................................... 73
  - Defining Command Line Options for Tools ...................................................................... 73
- Wise Package Studio Variables ............................................................................................ 75

### Chapter 4: Repackaging Applications and Managing Projects .................................................. 76
- About the Project and Tools tabs ............................................................................................ 76
- Using the Projects Tab ............................................................................................................ 76
- Using the Tools Tab ................................................................................................................ 78
- Connecting to a Web Application ........................................................................................... 79
- Managing Projects ................................................................................................................ 80
  - Entering Project Tracking Information ............................................................................ 80
  - Assigning Users to Tasks in a Project ............................................................................... 81
  - Entering Time for Tasks ..................................................................................................... 82
- Viewing Project Metrics ........................................................................................................ 83
- Creating a To-Do List ............................................................................................................ 84
- Workbench Reports ............................................................................................................... 85
  - Generating a Workbench Report ....................................................................................... 86

### Chapter 5: Wise Package Studio Tools ....................................................................................... 87
- About Wise Package Studio tools .......................................................................................... 87
- List of Wise Package Studio tools .......................................................................................... 87
- How Wise Package Studio tools interact with revision control ............................................ 88
- Application Isolation ............................................................................................................. 89
  - Creating a Package That Isolates .EXEs. ......................................................................... 90
  - Specifying OS Compatibility for Isolation ....................................................................... 93
  - Selecting Isolation Options ............................................................................................... 94
- ApplicationWatch .................................................................................................................. 94
  - ApplicationWatch Exclusion List ...................................................................................... 95
  - Creating a Package with ApplicationWatch ..................................................................... 95
- Command Line Builder ........................................................................................................ 97
  - Creating a Command Line With the Command Line Builder .................................... 98
    - Adding UI Options to Your Command Line ................................................................. 100
    - Adding Logging Options to Your Command Line ...................................................... 101
## Chapter 6: Package Validation

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>About Package Validation</td>
<td>145</td>
</tr>
<tr>
<td>Validating Installation Packages</td>
<td>146</td>
</tr>
<tr>
<td>About Customizing Validation Modules</td>
<td>148</td>
</tr>
<tr>
<td>Adding a Validation Module to Package Validation</td>
<td>148</td>
</tr>
<tr>
<td>Selecting Validation Rules to Use</td>
<td>149</td>
</tr>
<tr>
<td>About Rules That Call a Custom Action</td>
<td>150</td>
</tr>
<tr>
<td>Adding a Rule That Calls a Custom Action</td>
<td>151</td>
</tr>
<tr>
<td>About Validation Rule Sets</td>
<td>152</td>
</tr>
<tr>
<td>Adding a Validation Rule Set</td>
<td>153</td>
</tr>
</tbody>
</table>
Chapter 7: Test Expert

About Test Expert. 159
  Opening a Package in Test Expert. 160
  Setting Test Expert Preferences 161
  About the Master Test Plan 161
  Loading, Saving, and Clearing Results Files 163
  Installing an Installation Test into a Virtual Software Layer 164

About Test Cases 164
  Running a Test Case 165
  About Testing Groups of Packages 166
  Setting Test Statuses and Details 166
  Determining Your Test Environment 168
  Testing on Multiple Computers 168
  Machine Capture Settings 169
    Setting Directories to be Watched for Uninstall Tests 169
    Setting a File, Wildcard, or Directory to Be Ignored During Uninstall Tests 170
    Setting Registry Entries to be Ignored During Uninstall Tests 171
  Adding a User-Defined Test Case 172

Test Case Reference 172
  Installation Tests 173
    How to Run Installation Tests 174
    Launch Conditions Test Case 176
    OS Conditions Test Case 176
    Verify Installation Test Case 177
  Standard Tests 177
    Check Internet Connection 178
    Check Network Location 178
    Database Connectivity 179
    Execute Program 180
  Application Verification Tests 180
    Class IDs Test Case 181
    File Extensions Test Case 182
    Help Files Test Case 183
    ODBC Data Sources Test Case 184
    Prog IDs Test Case 184
    Search Locations Test Case 185
    Services Test Case 186
    Shortcuts Test Case 187
  Application Execution Tests 187
    How to Run Application Execution Tests 189
    Extra Files Test Case 190
    Extra Registry Entries Test Case 191
    File Coverage Test Case 191
    Isolated Files Test Case 192
    Registry Coverage Test Case 193
  Uninstall Tests 193
    How to Run Uninstall Tests 194
    Created Files Test Case 195
    Created Registry Entries Test Case 196
    Destroyed Files Test Case 197
## Chapter 8: Capturing Applications

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>About Capturing Applications</td>
<td>201</td>
</tr>
<tr>
<td>SetupCapture Configuration</td>
<td>202</td>
</tr>
<tr>
<td>Configuring Settings in SetupCapture Configuration</td>
<td>203</td>
</tr>
<tr>
<td>Selecting the Configuration File</td>
<td>205</td>
</tr>
<tr>
<td>Setting General Settings</td>
<td>205</td>
</tr>
<tr>
<td>Setting Directories to Watch</td>
<td>208</td>
</tr>
<tr>
<td>Exclusion List Guidelines</td>
<td>209</td>
</tr>
<tr>
<td>Building an Exclusion List Automatically</td>
<td>210</td>
</tr>
<tr>
<td>Setting File and Folder Exclusions</td>
<td>211</td>
</tr>
<tr>
<td>Setting a File to Be Excluded</td>
<td>211</td>
</tr>
<tr>
<td>Setting a Directory to Be Excluded</td>
<td>212</td>
</tr>
<tr>
<td>Setting a File to Be Excluded Based on a Wildcard</td>
<td>213</td>
</tr>
<tr>
<td>Converting User-Specific Files to Generic User Files</td>
<td>214</td>
</tr>
<tr>
<td>Setting Registry Exclusions</td>
<td>214</td>
</tr>
<tr>
<td>Setting INI File Exclusions</td>
<td>215</td>
</tr>
<tr>
<td>SetupCapture</td>
<td>216</td>
</tr>
<tr>
<td>Guidelines for Capturing an Installation</td>
<td>216</td>
</tr>
<tr>
<td>Setting Up a Clean Machine</td>
<td>218</td>
</tr>
<tr>
<td>Capturing an Installation in a Virtual Software Layer</td>
<td>219</td>
</tr>
<tr>
<td>Capturing an Installation</td>
<td>219</td>
</tr>
<tr>
<td>Specifying the Installation File</td>
<td>223</td>
</tr>
<tr>
<td>Configuring SetupCapture</td>
<td>224</td>
</tr>
<tr>
<td>Selecting the Capture Methodology</td>
<td>224</td>
</tr>
<tr>
<td>Selecting a Virtual OS File</td>
<td>226</td>
</tr>
<tr>
<td>Beginning the SetupCapture</td>
<td>226</td>
</tr>
<tr>
<td>Using a Previous Scan</td>
<td>227</td>
</tr>
<tr>
<td>Executing Installations to Be Captured</td>
<td>228</td>
</tr>
<tr>
<td>Editing SetupCapture Inclusions</td>
<td>228</td>
</tr>
<tr>
<td>Editing SetupCapture Exclusions</td>
<td>229</td>
</tr>
<tr>
<td>Finishing SetupCapture</td>
<td>230</td>
</tr>
<tr>
<td>Adding Merge Modules Instead of Files</td>
<td>232</td>
</tr>
<tr>
<td>Configuring the Installation as a New Feature</td>
<td>233</td>
</tr>
<tr>
<td>Using SetupCapture With Virtual Capture</td>
<td>237</td>
</tr>
<tr>
<td>Guidelines for Virtual Capture</td>
<td>238</td>
</tr>
<tr>
<td>Creating a Virtual OS</td>
<td>239</td>
</tr>
<tr>
<td>Using SetupCapture to Capture First Use Settings</td>
<td>240</td>
</tr>
<tr>
<td>SOE Snapshot</td>
<td>242</td>
</tr>
<tr>
<td>Guidelines for Capturing the Standard Operating Environment</td>
<td>243</td>
</tr>
<tr>
<td>SOE Snapshot Configuration Settings</td>
<td>243</td>
</tr>
<tr>
<td>Capturing the Standard Operating Environment</td>
<td>244</td>
</tr>
<tr>
<td>Capturing With Wise Web Capture</td>
<td>245</td>
</tr>
<tr>
<td>Files and Registry Entries Ignored During Captures</td>
<td>247</td>
</tr>
</tbody>
</table>

## Chapter 9: Package Distribution

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>About Package Distribution</td>
<td>249</td>
</tr>
<tr>
<td>Distribution Methods</td>
<td>250</td>
</tr>
<tr>
<td>Moving a Package into Microsoft Active Directory</td>
<td>250</td>
</tr>
<tr>
<td>Copying a Package to the Share Point Directory</td>
<td>252</td>
</tr>
<tr>
<td>Copying a Package to a Network Directory</td>
<td>254</td>
</tr>
</tbody>
</table>
Chapter 1
Introduction to Wise Package Studio

This chapter includes the following topics:

- About Wise Package Studio on page 12
- Repackaging Basics on page 14
- Starting Wise Package Studio on page 16
- The Workbench Interface on page 19
- About the Wise Software Repository on page 23
- Wise Package Studio Directories on page 24
- About the Share Point Directory on page 26
- Integration with Software Virtualization Solution on page 29
- Wise Package Studio File Types on page 31
- Wise Package Studio Status Types on page 33
- Product Documentation on page 33

About Wise Package Studio

Wise Package Studio® is a software packaging and application lifecycle management solution that supports the needs of application deployment and desktop management teams. Wise Package Studio provides a complete toolkit to support every phase of the application lifecycle, including application integration, package quality assurance, and release management. Use Wise Package Studio to:

- Improve the reliability of software installations, which reduces support costs and increases end user productivity. Wise Package Studio provides the processes and tools to effectively test an application prior to mass deployment.
- Enable faster and more reliable software rollouts by streamlining the process of preparing applications for distribution. Wise Package Studio accomplishes this through its project management tools, process automation, and built-in best practices.
- Support corporate standardization. The process-oriented approach to repackaging in Wise Package Studio helps standardize the repackaging process, while its complete editing capabilities let you customize the way software is installed.
- Achieve a greater return on your Windows operating system investment. Wise Package Studio provides complete capabilities for migrating applications to Windows Installer format, and customizing and validating Windows Installer packages.
- Manage every package used in your organization through each of its lifecycle phases, whether it is undergoing customization, in production, or retired.

See also:
Wise Package Studio Editions

Wise Package Studio is available in two editions, each designed to fulfill the needs of a particular type of user. The edition you purchase determines what features are available to you.

See Feature Summary on page 290.

Standard Edition is a stand-alone packaging tool for individuals who prefer an ad hoc approach to repacking. It provides Windows Installer packaging and validation functionality, helping organizations quickly and reliably migrate applications to the .MSI standard.

Professional Edition is an advanced packaging solution. It provides core functionality for advanced packaging, testing, and conflict management, and helps organizations support application standardization using a process-oriented approach. Professional Edition provides the starting point for adding extended functionality with the following modules:

- Enterprise Management Server is an enterprise application integration solution designed for packaging teams that may be either centralized or distributed. It provides advanced project management functionality and security, helping organizations create and manage a formalized, enterprise-wide application integration process.

- Quality Assurance covers all aspects of testing Windows Installer packages, including an easy way to conduct multiple testing activities in both the lab and real-world environments.

Wise Package Studio Terminology

Wise Package Studio
A collection of tools for managing the application lifecycle.

Workbench
The interface you use to work on repackaging projects or to run tools such as SetupCapture.

Project
Defines the job you need to accomplish. (Example: repackaging an application.) A project lets you track information about the job, such as status, dates created and modified, and notes. Using a project also lets you control the locations and names of the files that are created and used during the project. In the Professional Edition, you can associate a project with a process that defines the tasks to be performed.

Process
(Not available in Standard Edition.)
A list of tasks that you perform in order to complete a project. Wise Package Studio contains predefined processes and you can create new processes as needed.

Task
(Not available in Standard Edition.)
A single step to be performed in a process. A task can be associated with a Wise Package Studio tool or a third-party program. (Example: Microsoft Word or a drive imaging program.) Other tasks might not be associated with a tool or program, but might be something that you need to perform during the course of the process. (Examples: Establish clean machine, Install software.)

**Tool**

An executable application that you use to accomplish a task. Wise Package Studio includes predefined tools. In the Professional Edition, you can add new tools as needed.

**Installation**

- The compiled form of an installation, which is an .MSI or an .EXE.
- The project and source files that represent an installation that is created in a Wise development tool. Example: a .WSI or .WSE with source files.
- What happens on the destination computer when a package is opened.

**Package**

An application that is created, manipulated, or repackaged in Wise Package Studio. A package consists of:

- The distributable piece or pieces of an application (typically the installation file) and instructions for running the installation when it is deployed (typically a command line). A package might also contain additional files that should be distributed with the installation (example: an informational text file). This information represents a package definition.
- The source files associated with each package’s installation.

**Application**

A collection of similar packages in the Software Manager database. Example: Microsoft Word is an application; Word 2003 and Word 2007 would be packages of that application.

---

**Repackaging Basics**

Repackaging means changing or customizing a software installation to meet the needs of an organization. Repackaging is a critical step in the application lifecycle management that is supported by Wise Package Studio.

See *About Wise Package Studio* on page 12.

**Why Should You Repackage?**

- Create consistent and standardized, yet customized, installations. Repackaging an installation so that it adheres to your organization’s standards reduces the cost of supporting end users’ desktops.
- Create silent installations or limit the options available to end users. This streamlines installations and promotes ease of application deployment.
- Migrate installations to the Windows Installer format.
Many software installations are not in Windows Installer (.MSI) format. Repackaging those installations lets you take advantage of the Windows Installer features. In addition, Active Directory deployment requires .MSI format.

See *Advantages of the Windows Installer Format* on page 15.

**What Should Not Be Repackaged?**

Repackaging is not appropriate for certain types of applications:

- **.MSI files**
  
  Installations that are already in .MSI format should not be repackaged. Instead, use transforms to customize them. Transforms apply changes to the installation at run time to tailor the installed application to the needs of a particular group of users. For general information on transforms, see About Transforms in the Windows Installer Editor Help.

- **Windows Media Player, Microsoft Internet Explorer, antivirus software, and device drivers**
  
  These types of applications make low-level changes to the operating system involving Windows File Protection.

- **Distributable components of an operating system, including service packs, OS security updates, Internet Explorer, MDAC, or the Windows Installer service**
  
  These items are not repackaged because they break Windows security rules. The Windows Installer service might not run or might be modified by these installations. Service packs are not repackaged because it is difficult to capture all of the changes made to the operating system, and a significant number of service pack files are Microsoft file-protected. MDAC is not repackaged because it is a merge module.

**Advantages of the Windows Installer Format**

Wise Package Studio provides complete capabilities for migrating applications to Windows Installer format. Using Windows Installer results in a solid, robust installation that reduces the total cost of ownership and enables compliance with the Windows logo program. Because Windows Installer is part of the operating system, it provides benefits that are not available in traditional installation technology.

- **Installation rollback**
  
  If a Windows Installer installation fails, Windows Installer can return the computer to the precise state it was in before the installation. This includes restoring deleted or overwritten files, registry keys, and other resources.

- **Self-healing**
  
  (Also called automatic repair and self-repair.) Windows Installer can repair missing components of the application without rerunning the installation. When an application starts, Windows Installer checks a list of key files and registry entries. If it detects any problems, Windows Installer repairs the application using a cached database that contains key paths to application components.

- **Advertisement**
  
  (Also called install-on-demand and just-in-time installation.) Advertised features are not installed but appear installed to the user. Only the entry points for the features are installed. The first time a user invokes an advertised feature, it is installed.
• **Customization**
  You can customize the behavior of an installation by creating transforms. Transforms apply changes to the installation at run time to tailor the installed application to the needs of a particular group of users.

• **Componentization**
  Windows Installer uses components to group resources so they move as a unit. The installation database tracks which applications require a particular component, which files comprise each component, where each file is installed on the system, and where component sources are located.

• **Standardization**
  Windows Installer uses consistent and reliable version rules, which provide consistent and reliable installations for all applications and prevent newer files from being overwritten by older files. Windows Installer’s system-wide management of shared resources prevents conflicts that can occur when uninstalling one application removes files that are shared by other applications.

• **Elevated privileges**
  You can install or advertise applications by using system-level privileges regardless of the privileges of the user who is logged on to the computer.

• **Easier deployment of application updates**
  Windows Installer provides built-in patching technology to update installed versions of a Windows Installer-based application. Unlike full installations, a patch installation contains only the information necessary to update an installed version of the application.

  During an upgrade, Windows Installer detects whether the application to be upgraded was previously advertised or installed, and then removes it when installing the newer version. Additionally, Windows Installer allows for some migration of feature states from previously installed applications.

See also:

*Repackaging Basics* on page 14

---

**Starting Wise Package Studio**

**To start Wise Package Studio**

1. Select Start menu > Programs > Symantec > Wise Package Studio > Wise Package Studio.

2. If the Wise Package Studio Logon dialog box appears, log on as instructed by your Wise Package Studio administrator. (Not available in Standard Edition.)
   
   See *Wise Package Studio Logon Options* on page 17.
   
   If you cannot log on, one or more dialog boxes might appear.
   
   See *If Your Logon Fails*.

3. Click OK.

   The first time you start Wise Package Studio, Workbench opens to the Projects tab. The Standard Edition opens a project named Sample Project; the Professional Edition opens the Initial Workbench Setup project.
If Your Logon Fails

➤ Not available in Standard Edition.

You cannot log on to Wise Package Studio if:

- You have not configured the Wise Software Repository in the Wise Repository Manager. See Configuring the Wise Software Repository in the Getting Started Guide.

- You have not been assigned a Wise Package Studio license.
  - If a serial number is available, you might be assigned a serial number automatically. If not, the Assign User Licensing dialog box appears. Mark one or more check boxes for the licenses to assign.
    With Enterprise Management Server, you cannot be assigned a serial number automatically. The Wise Package Studio administrator must assign licenses.
  - If a serial number is not available, the Add Serial Number dialog box appears. See Adding Serial Numbers on page 49.
  - If you entered a user name from a Windows NT account, and Security Setup does not contain a security group that matches the domain group you belong to, you are prompted to contact your Wise Package Studio administrator. (Enterprise Management Server only.)

Wise Package Studio Logon Options

➤ Not available in Standard Edition.

When you start Wise Package Studio, the Wise Package Studio Logon dialog box appears. Obtain your logon information from your Wise Package Studio administrator.

Your entries in this dialog box depend on the type of logon account you use.

See Options on the Wise Package Studio Logon dialog box.

Types of Wise Package Studio logon accounts

<table>
<thead>
<tr>
<th>Logon account</th>
<th>Usage</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workbench account</td>
<td>This account is defined when you are assigned a Wise Package Studio license. Use it when:</td>
<td>None.</td>
</tr>
<tr>
<td></td>
<td>- You do not have an Enterprise Management Server license.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- The computer is not connected to a Windows NT domain. Example:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>When you use Wise Package Studio on a lab computer.</td>
<td></td>
</tr>
</tbody>
</table>
## Introduction to Wise Package Studio

### Options on the Wise Package Studio Logon dialog box

<table>
<thead>
<tr>
<th>Logon account</th>
<th>Usage</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current Windows NT account</td>
<td>Log on to Wise Package Studio as the currently logged-on Windows NT user.</td>
<td>• The computer must be connected to a Windows NT domain.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• You must have an Enterprise Management Server license.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Security Setup must contain a security group whose name matches a valid group in the NT domain, and you must be defined in that domain group. If you are in multiple NT groups, you are logged on under the first valid group that is encountered.</td>
</tr>
<tr>
<td>Windows NT account</td>
<td>Log on to Wise Package Studio with a Windows NT account. This account can be different from the one that is currently logged on to Windows. This lets you log on to Wise Package Studio from another computer, including a remote computer.</td>
<td>• The computer must be connected to a Windows NT domain.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The remote computer must have Wise Package Studio installed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• You must have an Enterprise Management Server license.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Security Setup must contain a security group whose name matches a valid group in the NT domain, and you must be defined in that domain group. If you are in multiple NT groups, you are logged on under the first valid group that is encountered.</td>
</tr>
</tbody>
</table>

### Option Workbench account entries

- **User Name**
  - (Professional Edition) Type your user name from User Licensing Setup.
  - (Enterprise Management Server) Type your user name from Security Setup.
    - Leave this box blank.
    - Type your Windows NT user name for the Windows NT domain.

- **Password**
  - (Enterprise Management Server only) Type your password from Security Setup.
    - Leave this box blank.
    - Type your password for the Windows NT domain.

- **Use Security From**
  - Click *(Workbench Database).*
    - Leave the default. This option is disabled when you select the next option.
    - Select the Windows NT domain name.

- **Always Use Current Network Login**
  - Uncheck this check box.
    - Check this check box.
      - When you start Wise Package Studio in the future, you are logged on automatically.
      - Uncheck this check box.
The Workbench Interface

When you start Wise Package Studio, you see the Workbench interface, from which you do most of your repackaging work. The left pane of Workbench contains the Project and Tools tabs, which you use to work on projects or run tools.

When you are working on a project, you use the Projects tab and select the project from **Active Project**. When you are not working on a project, use the Tools tab and double-click the tool’s icon.

The first time you start Wise Package Studio, it opens to the Projects tab. Thereafter, it opens to the last tab in which you worked and the last project you had open, if any. In the left pane of Workbench, you can switch between the Projects tab and Tools tab by clicking the appropriate tab or by using the shortcut keys Alt+P and Alt+T respectively.

The Description tab in the right pane displays help text about the active task or tool. Click a task or tool in the left pane to display its help. If a task is associated with a tool, you can toggle between the task help and the tool help. To do so, click the View Tool Help/View Task Help link in the upper right of the right pane. The right pane is visible only when you are in full screen mode.

For information about Workbench display modes, see *Resizing the Workbench Pane* on page 22.

The Projects Tab

On the Projects tab, you select a project from **Active Project**. The left pane of the Projects tab displays either tools or tasks, depending on whether the active project is associated with a process.

See *When a Project Has No Process* on page 20 and *When a Project Has a Process* on page 20.

The right pane of the Projects tab contains the following tabs:

- **Description**
  Displays help text for the active task or tool.

- **Details**
  Displays Project Setup information for the current project.

With Enterprise Management Server, additional tabs appear:

- **Project Management**
  Lets you record information about a project so you can manage it and track its progress. You can enter project information, enter time spent on each task, and assign users to specific tasks in a project. Users can run only the tasks that have been assigned to them.
  See *Managing Projects* on page 80.

- **Metrics**
  Displays a record of all events that have occurred for the current project. Information is recorded when a user works on a task or marks or clears a task’s check box. The Metrics tab can also contain notes about each event.
  See *Viewing Project Metrics* on page 83.
Introduction to Wise Package Studio

- **To Do**
  Displays a record of to-do items that have been created for the current project. To-do items are entered by users and represent actions that must be taken while working on the project.

  See *Creating a To-Do List* on page 84.

With Enterprise Management Server, Security Setup determines whether you have access to the Projects tab.

**When a Project Has No Process**

Projects can use an ad hoc approach that does not include a process. When a project does not have a process, the left pane of the Projects tab displays all available tools. To use a tool, click the Run link to the right of the tool. You do not have to use the tools in any particular order.

**When a Project Has a Process**

- *Not available in Standard Edition.)*

When a project is associated with a process, the left pane of the Projects tab displays the project’s tasks. When a task involves running a tool, you run the tool by clicking the Run link to the right of the tool. When you complete a task, you mark the task’s check box.
The Tools Tab

The Tools tab displays all available tools, organized in functional groups. These include the predefined tools and, in the Professional Edition, any tools you have added. When you click a tool name, the tool’s help text appears in the Description tab in the right pane. To run a tool, double-click the tool name.

With Enterprise Management Server, Security Setup determines whether you have access to the Tools tab.
Resizing the Workbench Pane

The toolbar contains resizing tools that let you resize the left Workbench pane and hide the right pane.

- **Full screen mode**
  Maximizes the Workbench window, displaying both the right and left panes. Workbench always opens in full screen mode. When you start a tool from this mode, Workbench changes to side by side mode unless you marked **Run all tools in Full Screen Mode** in Workbench Preferences. Workbench changes back to full screen mode when you close all Workbench tools.

- **Side by side mode**
  Hides the right pane and decreases the width of the left pane to make room for a tool window or dialog box. Workbench changes to side by side mode when you start a tool or run a task. The window size and position in this mode are retained from session to session.

- **Stay on top mode**
  Hides the right pane and changes the left pane to a small window that floats on top of all other windows. The window size and position in this mode are retained from session to session.
About the Wise Software Repository

Not available in Standard Edition.

The Wise Software Repository™ is a collection of software packages, resources and information about those resources, project management information, and quality assurance data used by organizations as part of the repackaging process. This scalable repository provides a centralized point for managing software packages at any stage of deployment.

The Wise Software Repository consists of:

- **Share point directory**
  Contains shared Wise Package Studio files and shared resources that are used to create Windows Installer installations. It also contains source files for packages in the Software Manager database. All Wise Software Repository databases are associated with a specific share point directory.

  See About the Share Point Directory on page 26.

- **Workbench database**
  Stores information that Wise Package Studio creates and uses. Examples: project, process, tool, and security information. A repository can contain only one Workbench database.

- **Software Manager database**
  Contains all software packages and other resources that are used by an organization. Other resources include: merge modules, device drivers, Group Policy Objects, and standard operating system environment snapshots. A repository can contain multiple Software Manager databases.

  See About the Software Manager Database in the Software Manager Help.

- **Wise Services database**
  (Formerly named Preflight database.) Stores the following data that is generated and used by Wise services:
  - Tasks that are managed by the Wise Task Manager. Examples: importing packages; running the Merge Module Wizard; compiling .MSI or .WSI packages in Software Manager; remotely compiling packages in Windows Installer Editor.
  - (Quality Assurance module only.) The results that are generated from deploying preflight packages, which are made with Package Instrumentation. These results are used by the Preflight Data Collector and Preflight Analysis Web applications.

  A repository can contain only one Wise Services database.

  See also:

  Wise Task Manager on page 141
  About Preflight Deployment on page 259

Multiple Repositories

- In a large enterprise with multiple teams, each team might use a different share point directory and Wise Software Repository. Because a Wise Package Studio server can be associated with only one active repository at a time, each team must install their repository on a different server.
A single Wise Package Studio server can contain multiple repositories. However, only one repository can be active at a time.

To change the active repository on a Wise Package Studio server, open the repository in the Wise Repository Manager.

A Wise Package Studio client can connect to any Wise Software Repository that it can access. To change a client’s default repository, use the Workbench Preferences dialog box > Repository tab and specify the share point that is associated with an active Wise Software Repository.

For configuration recommendations, see *Additional Wise Package Studio Configurations* in the *Getting Started Guide*.

**Benefits of Maintaining Package Information in the Wise Software Repository**

- Maintain a complete inventory of all applications used in your organization and store all packages and their source files in a centralized location.
- Manage each package throughout its lifecycle—from integration to testing and deployment through retirement.
- Maintain the status of each package in the repository and avoid problems typically caused by mixing production packages with those in development. Examples: accidentally deploying a package that is not ready for use, or unintentionally changing a proven production package.
- Reduce conflicts between applications before deployment, producing reliable, error-free deployments and reducing help desk calls.

When corporate developers have the Enterprise Edition of Wise for Windows Installer or Wise for Visual Studio .NET, they can use the Wise Software Repository to manage shared resources and ensure they always use the correct versions of shared resources.

**Wise Package Studio Directories**

Files that are used and created by Wise Package Studio are organized in several directories. In the Standard Edition, they are subdirectories of the Wise Package Studio application directory. In the Professional Edition, they are subdirectories of the share point directory, which lets multiple users share the files.

If your organization uses the Enterprise Edition of Wise for Windows Installer, the share point directory might contain additional information that is unique to that product.

The Wise Package Studio installation also contains subdirectories that are specific to Windows Installer Editor. For descriptions of those subdirectories, see *Installation Resources and Their Locations* in the Windows Installer Editor Help.

**Warning**

Do not edit or delete the contents of the Wise Package Studio directories outside Wise Package Studio or other Wise tools. Doing so will cause problems in Workbench, Software Manager, and ConflictManager and can result in loss of data.
### Directory Contents

<table>
<thead>
<tr>
<th>Directory</th>
<th>Contents</th>
</tr>
</thead>
</table>
| 000, 001, and so on (Not available in Standard Edition.) | Source files that are associated with each package's installation. The subdirectories are numbered sequentially. These subdirectories are created when:  
- You distribute a package to the share point directory.  
- You import a package into the Software Manager database, and you distribute source files.  
See *How Source Files Are Indexed* on page 27 and *About .QUE Files* in the Software Manager Help. |
| Available Packages (Not available in Standard Edition.) | Provides a centralized location for storing all packages that are available for deployment, keeping them separate from packages that are still in development. A separate subdirectory, named `application\package`, is created for each package. This subdirectory contains the compiled installation file, which is read-only, and any files needed for the installation, such as external .CAB files or SMS package definition files. This subdirectory is created when you change the package's status to Available. |
| Custom Actions | See *Installation Resources and Their Locations* in the Windows Installer Editor Help. |
| Languages | See *Installation Resources and Their Locations* in the Windows Installer Editor Help. |
| Merge Modules | See *Installation Resources and Their Locations* in the Windows Installer Editor Help. |
| Projects | Project information, including installation files, SetupCapture reports, and transform files. Information for each project is stored in a separate subdirectory. You define the subdirectory name when you create the project. The subdirectory is created the first time you open the project on the Projects tab. Package definition files (.WPF) and their defined files. Each package definition file is stored in its own subdirectory. This subdirectory has a Files subdirectory where the files specified in the package definition file are copied. |
| Reports (Not available in Standard Edition.) | The Report.ini file, which stores information about the predefined reports and, with Enterprise Management Server, any report customizations. The ReportConfig.ini file, which is used when saving a ConflictManager report directly to a file without it opening in a report viewer. |
| Resources | See *Installation Resources and Their Locations* in the Windows Installer Editor Help. |
| Scripts (Not available in Standard Edition.) | Temporary .QUE files representing packages that have been distributed but not imported into the Software Manager database. See *About .QUE Files* in the Software Manager Help. This directory also contains the package installation file when:  
- You distribute a package that is not part of a Workbench project.  
- You import a package into the Software Manager database.  
With Enterprise Management Server, the Scripts directory also stores information about package subscriptions. |
Introduction to Wise Package Studio

About the Share Point Directory

**Not available in Standard Edition.**

During the Wise Package Studio - Professional installation, you specify a share point directory. The default directory name is Wise Share Point, however, you can change the name at installation time.

For recommendations on where to locate the share point directory, see *Choosing the Location for the Share Point Directory* in the Getting Started Guide.

Each Wise Software Repository and its databases are associated with a unique share point directory. To change the default repository and its associated share point directory:

- (Client installations.) Use the Workbench Preferences dialog box > Repository tab.
- (Server installations.) Use the Wise Repository Manager.

**Share point directory contents**

- Workbench projects.
- Workbench tool and task help files.
- Installation files for the packages you create and work with in Wise Package Studio. Examples: .MSI, WSI, .WSE, .EXE, and so on.
- Resources that are used to create Windows Installer installations. Examples: installation templates, component rules, language files, and so on.
- Predefined Workbench reports.
Introduction to Wise Package Studio

- Temporary .QUE files representing packages that have been distributed but not imported into the Software Manager database.
- Source files of installations you import into the Software Manager database.
- Package definition files (.WPF) that are created with Package Definition and all of the files specified by the definition file.
- .INI files used to generate reports.
- Log files and .INI files associated with the tasks in Wise Task Manager.

See Wise Package Studio Directories on page 24.

If your organization uses the Enterprise Edition of Wise for Windows Installer, the share point directory might contain additional information that is unique to that product.

Deleting files from the share point directory

Warning
Do not edit or delete the contents of the Wise Package Studio directories outside Wise Package Studio or other Wise tools. Doing so will cause problems in Workbench, Software Manager, and ConflictManager and can result in loss of data.

A common question is "Can I clean up the share point by deleting unused source files?" The answer is no. It is too difficult to know which files are safe to delete. Also, ConflictManager lets you revert resolved packages to their original state, but if you delete a package's original files, you cannot revert.

The only recommended way to delete files from the share point directory is to delete the entire package from the Software Manager database. When you do so, you can delete the package's source files from the share point subdirectories (000, 001, and so on), if those files are not referenced by any other application. See Deleting a Package in the Software Manager Help.

How Source Files Are Indexed

Not available in Standard Edition.

A sequentially-numbered directory structure is created under the share point directory to store occurrences of installation source files when:

- You distribute a package to the share point directory.
- You import a single package or multiple packages to the Software Manager database, and you distribute source files.

An index file named wamdb.idx, located in the share point directory, records the location of the source files. Because files are indexed, distributing source files to the share point eliminates storage of duplicate files and results in smaller storage requirements than if you distribute to a network directory.

Example:

Suppose you have three packages, each containing a version of report.dll. The first time you distribute a package containing report.dll, the file is placed in the share point’s 000\001 directory. If you distribute another package containing the same version of report.dll, the file is not saved a second time, but a counter is set for that file in wamdb.idx. If you distribute a third package that uses a different version of report.dll,
the file is stored in a second directory, 000\002. The result is a set of all the unique source files used by all the packages in the Software Manager database.

Example: Populating the Share Point Subdirectories

➤ Not available in Standard Edition.

When you follow the process in a typical Windows Installer repackaging project, files and directories are created in the share point directory as follows.

<table>
<thead>
<tr>
<th>What You Do</th>
<th>What Happens</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create and open a project</td>
<td>A subdirectory for the project is created in the share point's Projects directory. Example: share point\Projects\Project_Name</td>
</tr>
<tr>
<td>Create the package installation file</td>
<td>The package installation file (.WSI) is placed in the Projects subdirectory.</td>
</tr>
<tr>
<td>Compile the package from within the process</td>
<td>The compiled file (.MSI or .EXE) is placed in the Projects subdirectory. This is considered a temporary installation.</td>
</tr>
</tbody>
</table>
| Distribute to the share point directory | - A numeric directory structure (000\001, and so on) is created to hold the installation’s source files.  
- A temporary .QUE file is created in the share point Scripts subdirectory. |
| Import to the Software Manager database | - The .QUE file is deleted.  
- Package and resource information is added to the Software Manager database. The installation file remains in the Projects directory and the Software Manager database references that location.  
- Paths in the installation file are changed to reference the new source locations (000\001, and so on). |
| Resolve conflicts | The resource information in the Software Manager database is changed. Example: You might change the package to use a newer version of report.dll. |
| Export from ConflictManager and recompile | The original installation in the Projects directory is changed. Example: The path is changed to refer to the newer version of report.dll. |
| Run a task to make the package available | - The package status is changed to Available.  
- A subdirectory for the package is created in the share point Available Packages directory. (Example: Available Packages\Application Name\Package Name\) The final compiled installation file and any other files needed for installation are placed in this subdirectory. The installation file is set to read-only. |

See also:

About the Share Point Directory on page 26
Integration with Software Virtualization Solution

Not available in Standard Edition.

About Software Virtualization Solution

Software Virtualization Solution (SVS) is a revolutionary approach to software management. By placing applications and data into managed units called virtual software packages, you can instantly activate, deactivate, or reset applications. Instead of running the installation of an application on a client computer, you simply deploy and activate a virtualized application. When an application needs repair, you reset it to its original state. It also lets you completely avoid conflicts between applications, without altering the base Windows installation. You can also host multiple versions of the same application on a computer, so that you can roll out and test a new version without removing the old version.

For more information about Software Virtualization Solution (SVS), search for Workspace Virtualization on the symantec.com Web site.

Working with virtual software packages in Wise Package Studio

You can use Wise Package Studio to create, edit, manage, and distribute virtual software packages. Wise Package Studio also incorporates the software virtualization technology into two of its tools to greatly enhance their capabilities.

See About Virtual Software Packages on page 30.

Several script actions in the WiseScript editing tools specifically deal with virtual software packages. You can use these script actions to manage and update the virtual software packages that you create.

In Wise Package Studio, you work with virtual software packages in the following formats:

- Virtual software layer
- Virtual software archive file (.VSA), which is a portable version of a virtual software layer
- Virtual software project file (.WVP), which is a project file that you compile to create a .VSA file

Software Virtualization Agent

The Software Virtualization Agent is installed when you install Wise Package Studio. You must have the agent installed to do the following:

- Use the Virtual Package Editor
- Import .VSA files into Software Manager
- Use SetupCapture to capture an application as a virtual software package
- Work in a virtual layer in SetupCapture and Test Expert.

The agent is not required to create SVS enabled packages in Software Manager. The agent must also be installed on any desktop computer on which you will use virtualized packages.
About Virtual Software Packages

➤ Not available in Standard Edition.

Creation of virtual software packages

In Wise Package Studio, you can use the following tools to create a virtual software package:

- **Software Manager**
  Software Manager lets you enable packages that are in the Wise Software Repository for the Software Virtualization Solution (SVS). You can enable .MSI packages or any type of package that has an associated package definition file. When an SVS enabled package is installed on a target computer where the Software Virtualization Agent is present, the enabled package can create a virtual software layer, install the package into the layer, and save and activate the layer. If the Software Virtualization Agent is not present, the package is installed normally. An SVS enabled package retains any configuration logic of the installation. (Example: If the original installation contains a prerequisite that checks for a specific version of the .NET Framework runtime and installs it if needed, this is retained in the enabled package.)

  See About SVS Enabled Packages in the Software Manager Help.

- **SetupCapture**
  SetupCapture lets you convert an existing .EXE or .MSI installation into a virtual software package. You can run SetupCapture from within Virtual Package Editor or as a stand-alone tool to capture the installation. You can then edit the package with Virtual Package Editor. A virtual software package created with SetupCapture does not retain any of the installation’s configuration logic.

  See About SetupCapture in the Virtual Package Editor Help.

- **Virtual Package Editor**
  Virtual Package Editor lets you edit any virtual software package. You can edit the contents of a layer, exclude data from a layer, and add deletion entries or data layers. You can also add command lines or WiseScripts to be executed when a layer’s events are triggered.

  See About Virtual Package Editor in the Virtual Package Editor Help.

  Virtual Package Editor also lets you create a new package as a virtual software layer. You can create an application layer or a data layer. When you create a layer, the output for that layer can be a virtual software layer or virtual software project file (.WVP). When you compile a .WVP file, it generates a .VSA file.

  See Wise Package Studio File Types on page 31.

Distribution and management of virtual software packages

After you create a virtual software package, you can use Package Distribution to distribute it to a network directory or an FTP server. You also can import virtual software packages (.WVP and .VSA) into Software Manager for purposes of impact and risk assessment.

Because there is no need to detect conflicts between a virtual software package and other packages, ConflictManager does not generate conflicts for these packages. By default, virtual software packages do not appear in ConflictManager, but you can change the conflict filtering options to display them.
The WiseScript Script Editor has a set of script actions that you can use to manage, edit, find, and create virtual software layers. Each of these script actions has SVS (Software Virtualization Solution) in its name. For your convenience, these script actions are grouped by default under the SVS Items title bar.

See About SVS Script Actions in the WiseScript Editor or WiseScript Package Editor Help.

**Workbench tools that use software virtualization technology**

- When you capture a package with SetupCapture using the SmartMonitor or Snapshot method, you can perform the capture in a virtual layer.
- In Test Expert, you can install and run a package in a virtual layer.

When you finish the capture or testing operations, you can use Symantec SVS applet to delete or deactivate the virtual layer and restore the computer to its original state.

See About the Altiris SVS Applet in the Virtual Package Editor Help.

### Wise Package Studio File Types

In Wise Package Studio, you can create and edit different types of Windows Installer and WiseScript installation packages. In addition, there are distinct file types for Windows Installer merge modules, patch files, transforms, and virtual software packages. Each extension is described below.

<table>
<thead>
<tr>
<th>Extension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.CAB</td>
<td>Cabinet file, which consists of multiple files compressed into one. A Windows Mobile device installation consists of a single, self-extracting .CAB file, which is generated by the CabWiz program from an information file (.INF).</td>
</tr>
<tr>
<td>.EXE</td>
<td>Installer file, either created by you or obtained from a software vendor. You create a WiseScript .EXE by compiling a WiseScript project file (.WSE) in WiseScript Editor or WiseScript Package Editor. You create a Windows Installer .EXE by using Windows Installer Editor to place an .MSI file inside an .EXE file. See Setting Build Options for a Release in the Windows Installer Editor Help.</td>
</tr>
<tr>
<td>.INF</td>
<td>Device information file, which specifies directories, files, settings, and configurations that are used to install a mobile device application. An .INF file is the project file format in the Mobile Device Package Editor, and is compiled to a .CAB file.</td>
</tr>
<tr>
<td>.LPR</td>
<td>(Not available in Standard Edition.) Linux project, which describes a Linux installation. You edit an .LPR in Linux Package Editor and compile it to a shell file (.SH).</td>
</tr>
<tr>
<td>.MSI</td>
<td>Windows Installer database, which is a distributable installation. The .MSI extension is associated with the Windows Installer executable, MSIEexec.EXE. When an .MSI is opened, Windows Installer executes it, thereby installing the application. You can open and edit an .MSI in Windows Installer Editor. However, options that have to do with creating an .MSI, such as those on the Releases, Release Settings, and Media pages, are unavailable.</td>
</tr>
<tr>
<td>Extension</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>.MSM</td>
<td>Windows Installer merge module, which is a pre-compiled library of components (files, registry changes, and other system changes) that installs a discrete part of your application. It cannot be run alone, but must be merged with an .MSI during the .MSI compile. See About Merge Modules in the Windows Installer Editor Help.</td>
</tr>
<tr>
<td>.MSP</td>
<td>Windows Installer patch, which updates an existing installed application. Patches contain only the differences between the old and new versions of an application. You create a patch installation with the Patch Creation tool, which creates an .MSP file that you distribute to end users.</td>
</tr>
<tr>
<td>.MST</td>
<td>Windows Installer transform, which changes a Windows Installer package at run time and must be applied from the command line. See About Transforms in the Windows Installer Editor Help.</td>
</tr>
<tr>
<td>.PCP</td>
<td>Windows Installer patch project, which describes and compiles to a Windows Installer patch. A .PCP file is created from the Patch Creation tool.</td>
</tr>
<tr>
<td>.SH</td>
<td>(Not available in Standard Edition.) Linux installation shell, which acts as a &quot;wrapper&quot; to the installation RPM. The shell file is compiled from the Linux project file (.LPR).</td>
</tr>
<tr>
<td>.SOE</td>
<td>(Not available in Standard Edition.) File created by SOE Snapshot when it captures a standard operating environment (SOE). You can import an .SOE into the Software Manager database to represent a baseline machine in your organization.</td>
</tr>
<tr>
<td>.VSA</td>
<td>(Not available in Standard Edition.) A virtual software archive file created by the Virtual Package Editor when you compile a .WVP file. A .VSA file is a portable virtual software package that becomes a virtual software layer when it is imported into Symantec SVS applet. A .VSA file is also created when you export a virtual software layer from Symantec SVS applet. See About the Altiris SVS Applet in the Virtual Package Editor Help.</td>
</tr>
<tr>
<td>.WOS</td>
<td>(Not available in Standard Edition.) A Virtual OS file created by the Virtual OS Creation utility.</td>
</tr>
<tr>
<td>.WPF</td>
<td>(Not available in Standard Edition.) A Wise package definition file (.WPF), which defines what is needed to install a package.</td>
</tr>
<tr>
<td>.WSE</td>
<td>(Not available in Standard Edition.) WiseScript project. This file extension is unique to Wise products. You can open and edit a .WSE in WiseScript Editor or WiseScript Package Editor and compile it to create a corresponding .EXE.</td>
</tr>
<tr>
<td>.WSI</td>
<td>Windows Installer project, which describes an .MSI but does not store contents. It is in the same format as an .MSI. You edit a .WSI in Windows Installer Editor and compile it to the corresponding .MSI. The .WSI file is smaller than an .MSI and you can set multiple options for the output of the .MSI.</td>
</tr>
<tr>
<td>.WSM</td>
<td>Windows Installer merge module project, which describes an .MSM, but does not store merge module contents. You edit a .WSM in Windows Installer Editor and compile it to the corresponding .MSM. See About Merge Modules in the Windows Installer Editor Help.</td>
</tr>
<tr>
<td>.WVP</td>
<td>(Not available in Standard Edition.) A virtual software project file that compiles to a .VSA file in the Virtual Package Editor. You can use SetupCapture to capture an application and save the output as a .WVP file. You can also create a .WVP file in the Virtual Package Editor.</td>
</tr>
</tbody>
</table>
Wise Package Studio Status Types

Use the different statuses in Wise Package Studio to manage your packages and projects.

**Project Status**
Indicates the state of projects in Workbench.

In the Standard and Professional Editions, a project can have either of two statuses: Open or Closed. To set the status, change the **Status** field in Project Setup.

With Enterprise Management Server, a project can have any of nine statuses: Open, Analyzing, Packaging, Validating, Testing, Deconflicting, Complete, Closed, and On Hold. Use the ones that meet your corporate standards. To set the project status:

- Select a status in the Project Management tab in Workbench, or
- Complete a task that has a status entered in the **Update project status upon task completion to** field in Process Templates Setup, or
- Change the **Status** field in Project Setup

Use the project status to filter projects in the Active Project list in Workbench. You set the filter criteria in Workbench Preferences.

**Package Status**
(Not available in Standard Edition.)
Indicates the state of packages in the Software Manager database. Use the package status to determine whether a package can be deployed to end users.

A package can have any of three statuses: Under Development, Available, or Retired. To set the package status, either change the **Status** field in the Software Manager Package pane or run Software Manager as a task that uses a command-line option to change the status automatically.

Use the package status to filter the display of packages in ConflictManager.

See *Changing the Package Status* in the Software Manager Help.

Product Documentation

This documentation assumes that you are proficient in the use of the Windows operating system. If you need help using the operating system, consult its user documentation.

Use the following sources of information to learn about this product.

**Online Help**
The online help contains detailed technical information and step-by-step instructions for performing common tasks.

Access help in the following ways:

- To display context-sensitive help for the active window or dialog box, press F1.
- To select a help topic from a table of contents, index, or search, select Help menu > Help Topics.
Reference Manuals

All the material in the online help is also available in a .PDF-format reference manual, which you can access by selecting Help menu > Reference Manual.

The following tools have separate manuals: ConflictManager, Linux Package Editor, Mobile Device Package Editor, Software Manager, Virtual Package Editor, Windows Installer Editor, and WiseScript Package Editor.

Getting Started

The Getting Started Guide contains system requirements, installation instructions, and a tutorial. You can access a .PDF version of the Getting Started Guide from the Windows Start menu.

The installation and repository management sections of the Getting Started Guide are also available as online help. In the Wise Repository Manager, select Help menu > Help Topics, or click the Help button on any of the Wise Package Studio installation dialog boxes.

Release Notes

The product release notes cover new features, enhancements, bug fixes, and known issues for the current version of this product. To access the release notes, select Release Notes from the Symantec program group on the Windows Start menu.

Windows Installer SDK Help (Windows Installer Editor only)

You can get technical details about Windows Installer from its own help system, which is written by Microsoft for a developer audience. In Wise for Windows Installer, select Help menu > Windows Installer SDK Help.

Version 4.5 of the Windows Installer SDK Help is provided. If you have obtained a later version, links from the Wise product documentation to the Windows Installer SDK Help might not work.
Chapter 2
Setting Up Wise Package Studio

This chapter includes the following topics:

- Steps for Setting Up Wise Package Studio on page 35
- About Wise Package Studio Security on page 38
- License Management on page 47
- Workbench Preferences on page 52

Steps for Setting Up Wise Package Studio

Setting up Wise Package Studio consists of defining your corporate repackaging standards and applying those standards to various settings and templates in Wise Package Studio and its tools.

Follow the steps below to set up Wise Package Studio. Depending on your organization's requirements, you might need to perform additional setup steps, or you might be able to skip some of these steps.

To perform the steps that require a Wise Package Studio tool, run the tool from the Tools tab or the Projects tab. In the Professional Edition, you can use the predefined project named Initial Workbench Setup, which contains tasks that help you perform these steps.

See Using the Initial Workbench Setup Project on page 38.

To set up Wise Package Studio

1. Define company standards.
   Define and document standards for repackaging applications.

   Examples:
   - The privileges under which applications are installed.
   - What you need to do to incorporate inventory or licensing control within your environment.
   - How you register files.
   - What file versions are approved.
   - How to handle duplicate files. For example, what do you want to do when you install an application, and a file in the installation already exists on the destination computer? Do you want to always overwrite if the file being installed is newer than the existing file, or always install all files regardless of version?

   If you do not already have corporate standards, you can use the Microsoft Windows application specification as a starting point. Search for "Application Specification for Microsoft Windows" in the MSDN Library (msdn.microsoft.com/library). After you define your standards, document them in a text processing program.

2. Configure licensing. (Not available in Standard Edition.)
Assign licenses that let users use Wise Package Studio.

To perform this step, select Edit menu > User Licensing. Add users, add licenses, and assign licenses.

See Assigning Licenses on page 50.

With Enterprise Management Server, you must add users in Security Setup.

See Creating Users on page 41.

3. **Configure security.** (Enterprise Management Server only.)

   Determine how your users can use Wise Package Studio.

   To perform this step, select Edit menu > Security to create security groups and assign permissions to each group. Then create users and assign them to groups. To provide an additional level of security, your database administrator can set permissions on tables in the Workbench and Software Manager databases.

   See About Wise Package Studio Security on page 38.

4. **Define SetupCapture standards.**

   Define standards for capturing installations with SetupCapture. Customize configuration settings that will be used each time SetupCapture is run.

   To perform this step, run SetupCapture Configuration.


5. **Capture standard operating environment.** (Not available in Standard Edition.)

   You can capture, or make a snapshot of, the standard operating environment (SOE) of a baseline computer. This lets you find conflicts between applications and the SOE.

   To perform this step, run SOE Snapshot and then use Software Manager to import the snapshot into the Software Manager database.

   See SOE Snapshot on page 242 and Package Import in the Software Manager Help.

---

**Warning**

Predefined templates are read-only. Editing them is not recommended, because they might be overwritten during Wise Package Studio upgrades. Instead, save customized templates with different names.

6. **Customize Windows Installer template.**

   By default, when you create a new Windows Installer package, Windows Installer Editor opens a project file configured with commonly-used default settings. This default project file is based on a template that you can customize.

   To perform this step, run Windows Installer Editor and open the file Windows Application.wsi, which typically is located in the Windows Installer Editor\Templates directory. Save the customized template with a new name.

   See Creating and Editing Installation Templates in the Windows Installer Editor Help.

7. **Customize merge module template.**

   By default, when you create a new Windows Installer merge module, Windows Installer Editor opens a file configured with commonly-used default settings. This default merge module file is based on a template that you can customize.
To perform this step, run Windows Installer Editor and open the file Merge Module.wsm, which typically is located in the Windows Installer Editor\Templates directory. Save the customized template with a new name.

See Creating and Editing Installation Templates in the Windows Installer Editor Help.

8. **Customize WiseScript template.** (Not available in Standard Edition.)

   By default, when you create a new WiseScript package, WiseScript Package Editor contains a basic installation script. This default script is based on a template that you can customize.

   To perform this step, run WiseScript Package Editor and open the file Empty Project.wse, which typically is located in the WiseScript Editor\Templates directory. Save the customized template with a new name.

   See Creating and Editing Installation Templates in the WiseScript Package Editor Help.

9. **Define ConflictManager settings.** (Not available in Standard Edition.)

   Define conflict settings, which determine the type of conflicts that are detected and the files and registry keys that are excluded from conflict detection. Also decide whether to use conflict resolution rules and, if so, decide which predefined rule sets to use.

   To perform this step, run ConflictManager. Select Setup menu > Conflict Settings and edit the default settings based on your corporate standards. Then select Setup menu > Conflict Resolution Rules and edit the predefined rule sets or create new rule sets (optional).

   See About Conflict Settings and Conflict Resolution Rules in the ConflictManager Help.

10. **Set preferences.**

    Set preferences to control the behavior of Workbench and several other tools.

    To perform this step, select Edit menu > Preferences in Wise Package Studio. Then set any options as needed.

    See Workbench Preferences on page 52.

    Also set preferences in:

    - Windows Installer Editor.
      See Setting Options in the Windows Installer Editor Help.
    - WiseScript Package Editor. (Not available in Standard Edition.)
      See Setting Preferences in the WiseScript Editor Help.
    - Software Manager. (Not available in Standard Edition.)
      See Setting Software Manager Preferences in the Software Manager Help.

When you complete these setup steps, you can begin to use Wise Package Studio.
Using the Initial Workbench Setup Project

➤ Not available in Standard Edition.

A predefined project named Initial Workbench Setup leads you through the setup of Wise Package Studio. Using this project ensures that you do not skip any important steps and helps you become familiar with the Workbench interface and its process-oriented approach.

The tasks in the Initial Workbench Setup project mirror the steps for setting up Wise Package Studio.

See Steps for Setting Up Wise Package Studio on page 35.

To set up Wise Package Studio

2. Click the Projects tab.
3. If the Initial Workbench Setup project does not appear, select it from Active Project.
4. Do the following for each task, in order:
   a. Select the task and read the help text that appears in the right pane. If the task is associated with a tool, you can display help text for the tool by clicking the View Tool Help link in the upper right of the right pane.
   b. If a Run link appears to the right of a task, it means that the task is associated with a tool. Click the Run link to start the tool, then use the tool as needed to perform the task.
   c. As you finish each task, mark the check box to the left of the task to indicate that the task is complete. Tasks that are associated with tools are set up to be marked complete automatically.

About Wise Package Studio Security

➤ Enterprise Management Server only.

You can set several levels of security to determine how your users can use Wise Package Studio.

Built-in security

Use Security Setup to control user access to Wise Package Studio tools and functions within tools. In Security Setup, you create security groups and assign permissions for each group.

See Creating Groups and Setting Permissions on page 39.

Examples:

- Restrict access to Process Templates Setup
- Prevent certain users from using the Tools tab
- Allow access to ConflictManager, but restrict access to ConflictManager functions such as resolving conflicts and deleting applications
After you create groups, you create users, assign licenses, and assign users to groups. The group assignment determines the user’s access to Wise Package Studio tools and functions within tools.

See Creating Users on page 41.

Database security

To provide an additional level of security, your database administrator can set permissions on tables in the Workbench and Software Manager databases. (Example: Setting certain tables to read-only prevents users from changing database tables to bypass the built-in security in Workbench.)

See Setting Database Security on page 46.

Project task access

You can assign users to specific project tasks. When users work on a project, they can perform only the tasks assigned to them.

See Assigning Users to Tasks in a Project on page 81.

Integrating With Windows NT Security

➤ The Workbench database must contain at least one Enterprise Management Server license.

In a Windows NT environment, you can integrate Wise Package Studio security with Windows security in several ways:

- During logon, a user can specify how to validate their logon by selecting from a list of Windows NT domains.
- A user can log on to Wise Package Studio with their current Windows NT user name. When the user starts Wise Package Studio, they are logged on to Wise Package Studio automatically.
- In Security Setup, instead of creating users individually, you can import an entire group of users from an NT domain. (Enterprise Management Server only.)

When a user logs on with a Windows NT account, or when they use the current network logon, Security Setup must contain a security group whose name matches a valid group in the NT domain, and the user must be defined in that domain group.

Recommendations:

- Set up Windows NT domain groups according to Wise Package Studio functions. Example: repackagers, managers, team leaders.
- A user should be in only one Package Studio-related NT group. If a user is in multiple NT groups, they are logged on under the first valid group encountered.

Creating Groups and Setting Permissions

➤ Enterprise Management Server only.

Use Security Setup to create and edit security groups. A security group consists of a group name and a series of permission settings. You use the settings to specify:
• Whether members of the group can view tabs and edit the project, process, and tool setups in Workbench.
• Which tools members of the group can use.
• Whether members of the group can use specific areas of Wise Package Studio. Example: Options under the Software Manager Settings folder allow access to specific functions in Software Manager.

Wise Package Studio contains three predefined security groups.
See *Predefined Security Groups* on page 41.

**To create groups and set permissions**

   The Security Setup dialog box appears.
2. Right-click in the left pane and select Add > Group.
   A new group appears in the list in the left pane and is selected in the right pane.
3. In **Name** in the right pane, type the name to use for this group.
   If you want users to log on with a Windows NT user name, Security Setup must contain a group whose name matches a valid group in the NT domain.
   See *Integrating With Windows NT Security* on page 39.
4. In Permissions:
   a. Mark options to allow access to areas of Wise Package Studio.
      ♦ **No Access**
         Members of this group cannot access the selected area.
      ♦ **View**
         Members of this group can display the selected area but cannot make changes.
      ♦ **Edit**
         Members of this group can add to or change information in the selected area.
   b. Under the Workbench Settings folder, mark check boxes to allow access to Workbench tabs. At least one of these check boxes must be marked.
   c. Mark check boxes under the following folders to allow access to specific functions in these tools:
      ♦ Software Manager Settings.
         See *Setting Software Manager and ConflictManager Security* on page 43.
      ♦ ConflictManager Settings.
         See *Setting Software Manager and ConflictManager Security* on page 43
      ♦ SetupCapture Configuration Settings.
         See *Setting SetupCapture Configuration Security* on page 44.
5. In the **Tools** list, mark the check boxes to allow access to individual tools. The list includes predefined tools as well as tools you create.

   The new group is saved when you create or select another group or user or when you click Close.
After you create a group, you can assign users to it.

See *Creating Users* on page 41.

**Predefined Security Groups**

➤ *Enterprise Management Server only.*

The following security groups are predefined and cannot be deleted.

**WPS Administrator**

This group has permissions for all options and cannot be changed. It contains one predefined user named Admin.

**Unassigned**

Users are added or moved to this group when:

- You add a user without assigning a license.
- You delete a group that contains users.
- You unassign all licenses from a user.
- A user has a license other than Professional Edition, Quality Assurance, or Enterprise Management Server.

Users in this group cannot log on to Wise Package Studio. You should reassign any users that are placed in the Unassigned group. To do so, you must assign them at least one license.

**Wise Users**

This group is reserved for users who have a license for Professional Edition or Quality Assurance, but not Enterprise Management Server.

Users are added or moved to this group when:

- You add a user with a license for Professional Edition or Quality Assurance.
- A user has a license for Professional Edition or Quality Assurance and Enterprise Management Server, and you delete or unassign the Enterprise license.

**Creating Users**

➤ *Enterprise Management Server only.*

Use Security Setup to create and edit users. A user has a user name, a password, and a security group assignment. The user name and password are used to log on to Wise Package Studio. The group assignment determines the user’s access to Wise Package Studio tools and functions within tools.

A quick way to add multiple users is to import users from an NT group.

Wise Package Studio has one predefined user, Admin, which is assigned to the WPS Administrator group. You can edit the Admin user but you cannot edit the WPS Administrator group.
To add a user

   The Security Setup dialog box appears.

2. In the left pane, right-click a group and select Add > User.
   The Assign User Licensing dialog box appears.

3. Mark one or more check boxes to assign licenses to the user and click OK.
   A new user appears in the list in the left pane and in the user entry fields in the right pane.
   If the Assign User Licensing dialog box indicates that no licenses are available, you must add serial numbers. Click OK. The user is added to the Unassigned group because users without a license cannot be added to any other group. Exit Security Setup and add one or more serial numbers.
   See Adding Serial Numbers on page 49.

4. In Name in the right pane, enter the name this user should use when logging on to Wise Package Studio. In a Windows NT environment, you can enter the user’s Windows NT user name.

5. In Password, enter a unique password for this user to use when logging on to Wise Package Studio. This should be different from the user’s Windows NT password, because you should not store NT passwords in the Workbench database.

6. In a Windows NT environment, you can enter the user’s domain in Domain. When the user starts Wise Package Studio while logged on to this domain, they are logged on to Wise Package Studio automatically.

7. From Group, select the security group to assign this user to.
   The new user is saved when you create or select another group or user or when you click Close.

To import users from an NT group

This creates users in the Workbench database with the same user names as users in the Windows NT group you import. This is different from when a user logs on with their domain logon.


If you import a Windows NT group multiple times, additions to the NT group are added to the Workbench database, however, removals from the NT group are not removed from the Workbench database.

   The Security Setup dialog box appears.

2. In the left pane, right-click a group and select Add > Import NT Group.
   The NT Group Import dialog box appears.

3. Complete the dialog box and click OK:
   - **Domain**
     Select the domain from which to import the users.
   - **NT Group**
     Select the domain group of users to import.
Setting Wise Package Studio

- **Assign serial numbers after import**
  Mark this to assign licenses to these users immediately. If you clear this check box, the users are added to the Unassigned group without license assignments.

4. If the Assign User Licensing dialog box appears, mark one or more check boxes for the licenses to assign to the users and click OK. Licenses are assigned to users in order.

- If there are not enough licenses, the users that do not get license assignments are added to the Unassigned group.
- Any user that is not assigned an Enterprise Management Server license is added to the Wise Users group.

### Setting Software Manager and ConflictManager Security

*Enterprise Management Server only.*

Security Setup contains settings that allow access to specific functions in Software Manager and ConflictManager.

**Recommended process**

1. Organize Software Manager and ConflictManager users into security groups according to the level of permissions you want to grant them.

   Examples:
   - Administrator. Usually the database administrator.
   - Workbench Users. Users who repackage applications and manage conflicts. You might create a separate group for team leaders, if you want to give them access to more functions than other users.
   - Management. Supervisors and managers who need to view conflicts, create groups, and view reports.

2. Create the groups in Security Setup.
   See Creating Groups and Setting Permissions on page 39.

3. Decide which areas of the ConflictManager interface to enable for each group. See the following tables for recommendations.

4. In the Security Setup dialog box:
   a. In the **Tools** list, mark **Software Manager** and **ConflictManager**.
   b. In the **Permissions** list, mark or clear check boxes under the Software Manager Settings and ConflictManager Settings folders.

#### Recommended security settings for Software Manager

<table>
<thead>
<tr>
<th>Mark This Check Box in Security Setup</th>
<th>Allows Access to</th>
<th>Set for Admin</th>
<th>Set for Leaders</th>
<th>Set for Users</th>
<th>Set for Managers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Administration</td>
<td>Delete database contents, compress database, edit Network Index Properties</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Setting Up Wise Package Studio

#### Recommended security settings for ConflictManager

<table>
<thead>
<tr>
<th>Mark This Check Box in Security Setup</th>
<th>Allows Access to</th>
<th>Set for Admin</th>
<th>Set for Leaders</th>
<th>Set for Users</th>
<th>Set for Managers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Package Subscriptions</td>
<td>Add and edit subscriptions; refresh subscriptions</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Group Setup</td>
<td>Create, edit, and delete package groups; set group properties; remove packages from groups</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Import Packages</td>
<td>Import packages</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Delete Packages</td>
<td>Delete packages from the database</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change Package Properties</td>
<td>Edit the Package Attributes dialog box (except meta data); create and edit package relationships</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Manage Meta Data Fields</td>
<td>Add and edit meta data fields</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Edit Meta Data Values</td>
<td>Edit meta data values in both Software Manager and ConflictManager</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Change Package Status</td>
<td>Change the Package Status in the Package pane</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add/Edit Package Definitions</td>
<td>Define packages using Package Definition</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

### Setting SetupCapture Configuration Security

> Enterprise Management Server only.

SetupCapture and SOE Snapshot use settings from a configuration file to determine certain aspects of the capture, such as what directories are examined for changes and what files and registry entries are excluded by default. When no security is set, users can use and edit any local configuration file or the configuration file in the share point directory.
In Security Setup, you can set permissions that govern the configuration file used for SetupCapture and SOE Snapshot. The permissions selectively enable and disable certain user interface items, such as buttons and options.

To access Security Setup, see Creating Groups and Setting Permissions on page 39. Mark the following check boxes under the SetupCapture Configuration Settings folder:

- **Allow Non-Shared Configuration File**
  Lets users select a file other than the one in the share point directory to use during SetupCapture and SOE Snapshot.
  Mark this check box to:
  - Let users select any local configuration file by clicking the Change button in SetupCapture.
  - Let users edit any local configuration file by clicking the Settings button in SetupCapture.
  - Let users edit any local configuration file in SetupCapture Configuration.
  Clear this check box to force users to use the configuration file on the share point for all SetupCaptures and SOE Snapshots.

- **Modify Configuration File on Share Point**
  Lets users edit the shared configuration file in the share point directory. Limit this permission if you plan to develop a global configuration file and maintain its integrity for all captures.
  Mark this check box to:
  - Let users select the configuration file in the share point directory by clicking the Change button in SetupCapture, and then edit it by clicking the Settings button.
  - Let users edit the configuration file in the share point directory in SetupCapture Configuration.
  Clear this check box to create a comprehensive configuration file in the share point directory and prevent changes to it.

**Items disabled by SetupCapture Configuration settings**

<table>
<thead>
<tr>
<th>Item</th>
<th>Where it Appears</th>
<th>When it is Disabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change button</td>
<td>Welcome page of SetupCapture, SetupCapture Configuration, and SOE Snapshot</td>
<td>Allow Non-Shared Configuration File is cleared</td>
</tr>
</tbody>
</table>
Setting up Wise Package Studio

Setting Database Security

To provide an additional level of security, your database administrator can set permissions on tables in the Workbench and Software Manager databases. This is not required, but is an option if you are concerned about unauthorized users changing database tables outside Wise Package Studio.

**Note**
The following recommendations assume that you are a database administrator familiar with creating and maintaining a SQL Server database. We do not offer technical support for SQL Server or SQL Server Express.

**Recommended database permissions**
The administrator should have read/write permission for all tables.

**Warning**
Setting permissions that are more strict than the following recommendations can result in database errors.
To set database security in SQL Server and SQL Server Express

Use SQL Server Enterprise Manager to set permissions on each table in the Workbench and Software Manager databases. You set permissions by either user or group, depending on whether you have set up database security groups. These groups are different from the groups in Security Setup in Wise Package Studio.

<table>
<thead>
<tr>
<th>Database</th>
<th>Tables</th>
<th>Permission to set for users other than administrator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workbench</td>
<td>GroupBits</td>
<td>Read-only</td>
</tr>
<tr>
<td></td>
<td>GroupTools</td>
<td>This prevents users from bypassing the built-in security in Wise Package Studio.</td>
</tr>
<tr>
<td></td>
<td>SecurityGroups</td>
<td></td>
</tr>
<tr>
<td></td>
<td>UserGroups</td>
<td></td>
</tr>
<tr>
<td>Workbench</td>
<td>UserPassword</td>
<td>Read/write if users can change their own passwords; otherwise read-only</td>
</tr>
<tr>
<td>Workbench</td>
<td>All other tables</td>
<td>Read/write</td>
</tr>
<tr>
<td>Software Manager</td>
<td>All tables</td>
<td>Read/write</td>
</tr>
</tbody>
</table>

License Management

Not available in Standard Edition

The flexible licensing model in Wise Package Studio lets organizations purchase the Wise Package Studio configuration that best meets their requirements. When you purchase Wise Package Studio, you receive one or more serial numbers. Each serial number represents one or more licenses for a specific edition, module, or bundle of Wise Package Studio.

Wise Package Studio is licensed per-user rather than per-machine. This means that a user can log on to Wise Package Studio from any computer that has Wise Package Studio installed.

Serial numbers and license assignments are stored in the Workbench database. Therefore, if a user selects a different share point directory (in Workbench Preferences), and thus a different Workbench database, that user must have a different license assignment in the Workbench database that they change to.

Users must be assigned a license for each edition and module of Wise Package Studio that they will use. Examples:

- Company A has a repackaging team of five. Only three people need to do repackaging. The other two people are quality assurance testers. This company purchases three licenses of Professional Edition and two licenses of Quality Assurance.

- Company B has a repackaging team of 10. All 10 need to do repackaging and conflict management. They require project management capabilities and user security. This company purchases 10 licenses of Professional Edition and 10 licenses of Enterprise Management Server.
Company B (the same company as above) decides to start doing quality assurance testing. They assign one person to do quality assurance in addition to their repackaging duties. This company purchases one license of Quality Assurance.

**Process for assigning licenses**

Following is an overview of the steps you take to assign licenses.

1. Add serial numbers to the Workbench database. The licenses that these serial numbers represent are considered available.
   
   See [*Adding Serial Numbers*](#) on page 49.

   
   See [*Creating Groups and Setting Permissions*](#) on page 39.

3. Add users and assign available licenses to users.
   
   See [*Assigning Licenses*](#) on page 50
   
   With Enterprise Management Server, also see [*Creating Users*](#) on page 41

Typically, you use User Licensing Setup to add and assign licenses. However, to facilitate the process, license assignments can also be made:

- When an unassigned user logs on to Wise Package Studio. (Professional Edition only. With Enterprise Management Server, the Wise Package Studio administrator must assign licenses.)
  
  The first time an unassigned user logs on to Wise Package Studio, if a license is available, they are prompted to select a license. If no license is available, they are asked to add a serial number. The user and license assignments are added to the Workbench database.

- In Security Setup, when you add a user or group of users. (Enterprise Management Server only.)

**About User Licensing Setup**

- *Not available in Standard Edition.*

User Licensing Setup provides a central location for managing Wise Package Studio licenses.

To access User Licensing Setup, select Edit menu > User Licensing.

In the User Licensing Setup dialog box, you can:

- Add serial numbers to the Workbench database.
  
  See [*Adding Serial Numbers*](#).

- Assign and unassign licenses to users.
  
  See [*Assigning Licenses*](#) on page 50.

- Delete serial numbers.
  
  See [*Deleting Serial Numbers*](#) on page 51.
Adding Serial Numbers

Not available in Standard Edition.

Before you can assign licenses to users, you must add serial numbers to the Workbench database. Use the Add Serial Number dialog box, which appears:

- During logon, if the user logging on has not been assigned a serial number, and no serial numbers are available. (Professional Edition only. With Enterprise Management Server, the Wise Package Studio administrator must assign licenses.)
- During logon to an evaluation version of Wise Package Studio, if the user clicks the Add Production Serial Number button on the Evaluation Central dialog box or the Evaluation Notice dialog box.
  
  See About Evaluation Serial Numbers.
- In User Licensing Setup.
  Select Edit menu > User Licensing. On the User Licensing Setup dialog box, click Add Serial Numbers.

This is the easiest way to enter multiple serial numbers. Do this when you first set up Wise Package Studio, when you purchase additional licenses, or when you purchase upgrades.

Note
You cannot add production serial numbers to an evaluation version of Wise Package Studio from User Licensing Setup. Add them from the Evaluation Central dialog box, which appears when you log on to an evaluation version.

On the Add Serial Number dialog box, you can import a license file containing multiple licenses. A license file is a text file with the extension .WLC and the following format:

serial number=user name.

Example:

XXXX-XXXX-XXXX-XXXX=maryk

The user name is optional; however, if it is included, the serial number assignment is made when you import the file. With Enterprise Management Server, the users are added to the Unassigned group.

To add serial numbers

1. Access the Add Serial Number dialog box as described above.
2. On the Add Serial Number dialog box, do either of the following:
   - In the Serial Number field, enter a 16-character serial number and click Add.
   - Click Browse and select a .WLC file.

   The serial numbers you add are listed on the dialog box. If you import a .WLC file that contains license assignments, the assigned users are listed also.

3. If you entered an upgrade serial number, the Previous Serial Number dialog box appears. Enter the serial number of the previous version and click OK.
4. Click OK on the Add Serial Number dialog box.

   If you added a serial number for an edition or module that contains a Web application (Professional Edition, Quality Assurance, or Enterprise Management...
Setting Up Wise Package Studio

(Enterprise Management Server), you must install the Web application. See *Installing Web Applications* in the *Wise Package Studio Getting Started Guide*.

### About Evaluation Serial Numbers

An evaluation serial number cannot be added to a database that contains production serial numbers.

An evaluation serial number expires when the evaluation time period elapses. When you log on with an expired serial number, the Evaluation Notice dialog box appears.

- If a production serial number is available, you are assigned a serial number and logged on.
- If a serial number is not available, the Add Serial Number dialog box appears.

You cannot add production serial numbers to an evaluation version of Wise Package Studio from User Licensing Setup. Add them from the Wise Package Studio Evaluation dialog box, which appears when you log on to an evaluation version.

When you add a production serial number to an evaluation version, all evaluation licenses are deleted.

### Assigning Licenses

> *Not available in Standard Edition.*

Users must be assigned a license for each edition or module of Wise Package Studio they will use. You assign licenses to users in User Licensing Setup. With Enterprise Management Server, you also can assign licenses in Security Setup.

Before you can assign licenses to users, you must add serial numbers to the Workbench database.

See *Adding Serial Numbers* on page 49.

With Enterprise Management Server, you must add users in Security Setup.

See *Creating Users* on page 41.

#### To assign licenses in User Licensing Setup

1. Select Edit menu > User Licensing.
   
   The User Licensing Setup dialog box appears. The upper pane lists the licenses in the current Workbench database, and the lower pane lists the licenses that are available to be assigned.

2. Click Assign Licenses.

3. In the upper pane of the Assign Licenses dialog box, select one or more users.

   If the user you want is not listed, click Add User and enter the user name.

   When you select a user, that user’s existing license assignments appear as marked check boxes in the lower pane. When you select multiple users, and some of the selected users have a particular license assignment, that check box is marked and shaded.

4. In the lower pane, mark one or more check boxes for the licenses to assign to the selected users.
To unassign a license, clear the check box. When you unassign a license, the user might be moved to a different group or the Unassigned group (Enterprise Management Server only).

See Predefined Security Groups on page 41.

5. When you finish assigning licenses, click OK.

6. With Enterprise Management Server, you might need to move the new user to another group in Security Setup after you assign licenses. Example: When you add a user without assigning a license, the user is added to the Unassigned group. After you assign a license, you can move the user to a different group.

To assign licenses in Security Setup

Enterprise Management Server only.

   The Security Setup dialog box appears.

2. In the left pane, right-click a user and select Modify License.
   The Assign User Licensing dialog box appears.

3. Mark one or more check boxes to assign licenses to the user and click OK.

4. You might need to move the new user to another group in Security Setup after you assign licenses. Example: When you add a user without assigning a license, the user is added to the Unassigned group. After you assign a license, you can move the user to a different group.

Deleting Serial Numbers

Not available in Standard Edition.

You can delete serial numbers from the Workbench database, which means they will not be available for assignment to users. Example: If you are converting an evaluation version of Wise Package Studio to a production version, you can delete any evaluation serial numbers.

To delete serial numbers

1. Select Edit menu > User Licensing.
   The User Licensing Setup dialog box appears.

2. Click Delete Serial Numbers.
   The Delete Serial Number dialog box appears.

3. Select one or more serial numbers to delete and click OK. The serial number is deleted.

4. Click Close on the User Licensing Setup dialog box.

If the deleted serial number was assigned to a user, that user might be moved to a different group or the Unassigned group. (Enterprise Management Server only.)

Example:

Suppose a user has licenses for Professional + Enterprise Management Server + Quality Assurance.
• If you delete the Quality Assurance serial number, the user is not moved to another group.

• If you delete the Enterprise serial number, the user is moved to the Wise Users group.

See Predefined Security Groups on page 41.

Workbench Preferences

Set preferences to control the behavior of Workbench. All preference settings affect only the current user on the local computer.

Select Edit menu > Preferences. Complete the General tab (described below), or click another tab for:

• Activating Suppressed Prompts on page 53
• Setting Repository Preferences on page 53

Setting General Preferences

• Active Project Filter
  Specify which projects appear on the Projects tab. You can display all projects or only open projects. With Enterprise Management Server, two additional options let you display all projects in which the next task is assigned to you, or all projects containing a task assigned to you.

• Run all tools in Full Screen Mode
  This does not apply to tools that have wizard interfaces.

• Create backup copy during save
  Mark this to have a new backup file created every time you save a file in either Windows Installer Editor or WiseScript Package Editor. This check box also appears in the preferences for each editor; this global setting affects the settings in the individual editors.

  The backup file name consists of the current file name plus a number. (Example: if the current file name is Sample.wsi, the backups are named Sample1.wsi, Sample2.wsi, and so on.) Only the file you are working on is backed up. (Example: if you open a .WSI and save it, the corresponding .MSI is not backed up.) Use caution with this option if you are working with large installation files; if you save often, your disk space will quickly become depleted.

• Enter actual hours by
  (Enterprise Management Server only.) Mark an option to specify whether you will enter hours completed on the Project Management tab for the entire project or for individual tasks.

• Allow Connection Attempts to Web Applications
  When you run a Web application tool, and Workbench cannot connect to the computer that is hosting Web application, the Connection Failed dialog box appears and might continue to appear as you use Workbench.

  Mark this check box to allow connection attempts and activate the Connection Failed dialog box. Clear it to prevent connection attempts and disable the dialog box.

  If you mark the Prevent Connection Attempts to Web Applications check box on the Connection Failed dialog box, this check box is cleared.
See *Connecting to a Web Application* on page 79.

### Activating Suppressed Prompts

To reactivate prompts that you previously suppressed, select Edit menu > Preferences. On the Workbench Preferences dialog box, click the Prompts tab.

Example: If an alert dialog box had a check box labeled **Don’t show this message again**, and you marked it, the prompt would appear here.

To reactivate a prompt, select it and click Activate.

### Setting Repository Preferences

> *Not available in Standard Edition.*

(Client installations only.) Use the Repository tab on the Workbench Preferences dialog box to connect to a different Wise Software Repository (the share point directory and any databases associated with it).

**Note**

Changing the share point directory or a specific path does not copy resources to the new location. Typically, you will specify a share point or path that is already in use.

**To connect to a different repository**

(Client installations only.) You can connect to a different repository by specifying the share point that is associated with it.

When you change the default share point, you are logged off and prompted to log on. Because serial numbers and license assignments are stored in the Workbench database, you must have a different license assignment in the Workbench database that you change to.

1. Select Edit menu > Preferences.
2. On the Workbench Preferences dialog box, click the Repository tab.
3. Click Browse.
4. On the Browse for Folder dialog box, browse to an existing share point directory and click OK.

The Wise Software Repository that is associated with that share point becomes your default.

To change the default Wise Software Repository for a server installation, use the Wise Repository Manager. See *Managing the Wise Software Repository* in the *Getting Started Guide.*
Chapter 3
Creating Projects, Processes, and Tools

This chapter includes the following topics:

- About Projects on page 54
- About Process Templates and Tasks on page 58
- About Tool Setup on page 66
- Help for Tasks and Tools on page 70
- Command Line Options on page 72
- Wise Package Studio Variables on page 75

About Projects

Wise Package Studio is project-oriented. A project defines the job you need to accomplish. (Example: repackaging an application.) Each project lets you record information about that job, including where project files are stored and what the project files are named. This provides greater control over the project files and simplifies the job for users.

Project creation

Use Project Setup to create a project for each new job. When you create a project, you enter information about the project and specify where project files are stored and how project files are named. With Enterprise Management Server, Security Setup determines whether you have access to Project Setup.

See Adding a New Project on page 55.

Project editing

The predefined projects cannot be edited.

With Enterprise Management Server, each project has an owner. The owner can be a user, a group, or none. When you assign an owner to a project, you limit who can edit the project.

Project process

In the Professional Edition, a project can be associated with a process. A process is a series of tasks that, when completed, result in a repackaged software installation. Using a process provides consistency in your repackaging and ensures that the project is performed according to corporate standards. When you create a project and assign a process to it, a copy of the original process template is stored with the project. You can change the process within a specific project without changing the original process template.

See About Process Templates and Tasks on page 58.

With Enterprise Management Server, you can create a project that uses a process from an external database. If the external process uses any user-defined Workbench tools or
command-line options that are not already in your database, the tool and command-line definitions are copied to your database. Changes to an external process template do not affect existing local projects created with that template.

See *External Process Templates* on page 60.

**Project usage**

To work on a project, you use the Projects tab.

See *Using the Projects Tab* on page 76.

**Project information as variables**

The information stored with the project provides values for the Wise Package Studio variables. (Example: The *File Name* field in Project Setup provides the value for the variable `[FileName]`.)

See *Wise Package Studio Variables* on page 75.

### Adding a New Project

**To add a new project**

1. Select Edit menu > Project.
   
   The Project Setup dialog box appears.

2. Right-click in the left pane and select Add.
   
   A new project appears in the project list in the left pane.

3. In the project entry fields in the right pane, specify the following:

   - **Project Name**
     
     Enter a unique name for this project. Do not use the following special characters: `/ \ * < > |`

   - **Project Directory**
     
     Specify the directory in which to store all files associated with this project. If multiple users will work on this project, enter a network directory. You can add Wise Package Studio variables to the path.

     See *Wise Package Studio Variables* on page 75.

     The default project directory is `\Sharepoint\Projects\[ProjectName]`. If you change the default project directory, your changes are saved and used the next time you create a project. Example: If you change the project directory to `\Projects`, the next time you create a new project, the project directory will be `\Projects\[ProjectName]`.

   - **Status**
     
     Set project status.

     In the Standard and Professional Editions, a project can be Open or Closed. The only way to change project status in those editions is in this field.

     With Enterprise Management Server, a project can have any of nine statuses. Typically, you should use the Project Management tab to maintain project status.

     See *Managing Projects* on page 80.
When you change the status in Project Setup, it is changed in the Project Management tab as well, and vice versa. When a project has no process, the Project Management tab is unavailable and you must change the status here.

- **Product Vendor**
  Specify the company that produces the application.

- **Application Name**
  Enter the name of the application. If your company is using Software Manager, this name will be used to identify this installation package in the Software Manager database. If the Software Manager database already contains a version of this application, enter the name exactly as it appears there. If this is a new application, enter a unique name. If this project creates a Windows Installer package, it will have this application name when it is opened in Windows Installer Editor (on the Product Details page) and Software Manager.

- **Package Name**
  Enter a package name to distinguish this installation from others with the same application name. A package can represent a version of a single application or a component of a larger suite. (Example: The Adobe Acrobat application might contain two packages, Acrobat 8.0 and Acrobat 9.0. The Microsoft Office application might contain packages named Office 2003 and Office 2007.) If your company is using Software Manager, this name will be used to identify this installation package in the Software Manager database. If this project creates a Windows Installer package, it will have this package name when it is opened in Windows Installer Editor (on the Product Details page) and Software Manager.

- **File Name**
  Enter the name to use for all files that are created and used by the tasks in this project. An appropriate extension is appended to each type of file. (Example: If you enter Sample here and run SetupCapture, a file named Sample.wsi is created.) If you give a project a file name and directory that matches that of an existing project, when you try to close the Project Setup dialog box you will receive a warning that you must change one of those values.

- **Vendor Package**
  Specify the name and location of the compiled executable (.EXE) or .MSI that you are going to repackage. You can use a variable to represent the path. If the package is in either the project directory or the share point directory, the [ProjectDir] or [Sharepoint] variable automatically replaces that portion of the directory when you browse to select a vendor package.

- **Project Owner**
  (Enterprise Management Server only.) Click and select an owner for the project from the Select Owner dialog box. The groups and users that appear in the Select Owner dialog box are created in Security Setup. See Creating Groups and Setting Permissions on page 39.

  The owner can be a user, a group, or none. When you assign an owner to a project, you limit who can edit the project.

<table>
<thead>
<tr>
<th>Owner</th>
<th>Who Can Edit the Project</th>
</tr>
</thead>
<tbody>
<tr>
<td>user</td>
<td>That user and anyone in the WPS Administrator group</td>
</tr>
<tr>
<td>group</td>
<td>Any user in that group or the WPS Administrator group</td>
</tr>
<tr>
<td>none</td>
<td>Any user</td>
</tr>
</tbody>
</table>
In an organization with mixed Professional and Enterprise Management Server licenses, when a user with a Professional Edition license creates a project, the **Project Owner** field does not appear. However, if a user with an Enterprise Management Server license opens that project, the **Project Owner** field appears and defaults to the Admin user.

- **Process**
  
  (Not available in Standard Edition.) Select the process to associate with this project. With Enterprise Management Server, if you are connected to an external database, this list includes processes from the external database. External processes have (External) appended to their names.

  When you select a process, a copy of the original process template is stored with the project. If you select an external process, and you reopen the project later, the process does not have (External) appended to its name anymore because the project contains a copy of the external process.

  To change the process for this project only, click Edit to the right of **Process**. Making changes here does not affect the original process template or any other projects that use the same process template.

  **Warning**
  
  If you change the process or select a different process after work on this project has begun, any existing task data will be lost.

- **Notes**
  
  Enter any additional information about the project.

  4. To undo all changes you made to the new project, right-click the project in the left pane and select Revert. You cannot revert after clicking another project.

  5. Save the new project by clicking Close or by creating or selecting another project on the Project Setup dialog box.

See also:

*External Process Templates* on page 60

### Duplicating or Deleting a Project

Use Project Setup to duplicate or delete a project. Access Project Setup by selecting Edit menu > Project.

With Enterprise Management Server, Security Setup determines whether you have access to Project Setup.

**To duplicate a project**

In Project Setup, right-click a project and select Duplicate. A copy of the project appears at the end of the project list. Tasks are copied from the duplicated project, not from the original process template. The new project is saved when you click Close or click another project.

See *Adding a New Project* on page 55.
To delete a project

In Project Setup, right-click a project and select Delete. You cannot undo project deletion. Deleting a project removes the project record from the Workbench database. It does not delete any files, such as installation files, that are related to the project.

About Process Templates and Tasks

➤ Not available in Standard Edition.

What is a process?

A process is a series of tasks you perform to complete a project. A process provides a logical, consistent approach to repackaging. You can set up a process and associate it with any number of projects. This saves time, reduces training requirements, and lets you apply a consistent methodology to similar projects.

With Enterprise Management Server, each process has an owner group. Only members of the owner group or the WPS Administrator group can edit a process.

What is a task?

A task is a single step to be performed in a process. A task can be associated with a Workbench tool or a third-party program. (Example: Microsoft Word or a drive imaging program.) Other tasks might not be associated with a tool or program, but might be something that you need to perform during the course of the process. You can organize tasks into parent tasks and subtasks.

About predefined processes

Wise Package Studio contains predefined processes that are based on industry best practices. View the predefined processes to see if they meet your needs. If not, you can duplicate one and customize the copy for your organization, or you can create a new process.

See Predefined Process Templates on page 59.

About creating and editing processes

Use Process Templates Setup to create and edit processes and tasks. When you create a new process, you add tasks and organize them into a logical order. You can also duplicate a process, copy tasks between processes, export processes to files, import process files, and delete processes and tasks. Changing a process template does not affect existing projects that use the template.

When you create a project and associate a process with it, a copy of the original process template is stored with the project. You can change the process within a specific project without changing the original process template.

See Adding a New Project on page 55.

The Process Templates Setup Interface

➤ Not available in Standard Edition.

You create processes and tasks in Process Templates Setup. To access this, select Edit menu > Process Templates.
At the top of the left pane of Process Templates Setup is a drop-down list that lets you filter the process list to view all processes, only predefined processes, or only user-defined processes.

(Enterprise Management Server only) If you are connected to an external database, you also see processes from the external database, which have (External) appended to their names.

In the process list in the left pane of Process Templates Setup, processes and their tasks are displayed in a tree structure. Expand and collapse processes to view tasks and subtasks by using the Expand, Expand All, and Collapse commands from the right-click menu.

Click a process or task in the left pane to display its detail in the right pane. This is where you define the process and its tasks. The right pane also contains a text editor, where you can create or link to a help file for tasks you define.

See also:
External Process Templates on page 60

Predefined Process Templates

Not available in Standard Edition.

Wise Package Studio contains predefined processes that are based on industry best practices. To see the predefined process templates, select Edit menu > Process Templates.

Predefined process template names are gray because they cannot be changed. This allows for updates and enhancements to the predefined process templates in future releases. However, you can duplicate them and customize the copies. You can also copy tasks from predefined process templates to processes you create.

The predefined process templates are:

Workbench configuration
This process is associated with the project Initial Workbench Setup, which leads you step-by-step through the setup of your Wise Package Studio environment. Normally, you only need to perform this process once, when you first install the product. You might repeat this process if your corporate standards change.

See Using the Initial Workbench Setup Project on page 38.

Repackage for Windows Installer
Leads you through the steps needed to repackage an installation as a Windows Installer package.

This process works with a single .MSI having the default project file name. If you create multiple releases for a package, you should either customize the process for that project to perform tasks on all .MSIs that are compiled, or write a macro to change the names of the additional .MSIs to the default file name.
Creating Projects, Processes, and Tools

For details, click the Workbench Projects tab and click a task in the left pane to display its help.

**External Process Templates**

- **Repackage into .VSA format**: Leads you through the steps needed to repackage an installation as a Virtual Software Package. This process creates a virtual software archive file (.VSA).
- **Repackage using WiseScript**: Leads you through the steps needed to repackage an installation as a WiseScript.
- **Customize MSI using transform**: Customizes an existing Windows Installer package by creating a transform file. This is typically done to eliminate end user interaction during the installation. The results of this process are a transform file and a shortcut to apply the transform to the base Windows Installer package.
- **Installation Quality Assurance**: Leads you through the steps needed to test a Windows Installer package. (Quality Assurance only.)

You can maintain a separate, or external Wise Software Repository whose Workbench database contains master process templates that you have customized to meet your organization’s standards. You then connect to that database from one or more other repositories so that users of those repositories have access to the master templates. This ensures that all users across your organization use the same standard, approved processes.

Use Wise Repository Manager to connect to an external database by selecting the share point directory with which it is associated. See Connecting to an External Workbench Database in the Getting Started Guide.

When the connection is made, the process templates in that database become visible in Workbench, and the predefined process templates in the local database become unavailable. If the external share point is disconnected or otherwise unavailable, then the predefined process templates in the local repository become available. Process templates that users create in their local database are always available.
How external processes appear in Process Templates Setup

Adding a New Process

➤ Not available in Standard Edition.

If the predefined processes do not meet your needs or if you require additional processes, you can use Process Templates Setup to create a new process. Part of creating a process is adding and arranging tasks.

With Enterprise Management Server:

- Security Setup determines whether you have access to Process Templates Setup.
- You can create process templates in a master database, to be used by multiple repackaging teams across your organization.
- You can create processes in a regional database, to be used only by members of a specific team.

To add a new process

1. Select Edit menu > Process Templates.
   The Process Templates Setup dialog box appears.
2. Right-click in the left pane and select Add > Process.
   A new process appears in the list on the left, with process entry fields on the right.
3. In Process Name, type the name for this process.
4. In Process Notes, enter a brief description.
5. (Enterprise Management Server only.) In Owner Group, specify a security group to restrict editing of this process. Only members of the owner group or the WPS Administrator group can edit this process. If you specify None, then anyone can edit this process.
6. Add tasks to this new process.
   See Adding Tasks to a Process on page 62.
7. Save the new process by clicking Close or by clicking another process or task in the Process Templates Setup dialog box.

You can rearrange tasks after creating them.

See Organizing Tasks and Processes on page 66.

Adding Tasks to a Process

Not available in Standard Edition.

Tasks are activities that must be performed to complete a process. Tasks can run Workbench tools, installed applications such as Microsoft Word, or any other executable programs. Tasks can also be manual, meaning they represent activities, such as analyzing the vendor package, that do not require a program to be run. Header tasks do not represent things the user has to do, instead, they act as informational headings for sets of subtasks.

Use Process Templates Setup to add tasks to a process and associate them with tools or other programs as needed. The tasks in a process should be listed in the order in which they must be performed. You can rearrange tasks after creating them.

See Organizing Tasks and Processes on page 66.

With Enterprise Management Server, Security Setup determines whether you have access to Process Templates Setup.

Note
You can copy a task from any process into processes that you create. Because predefined processes are read-only, you cannot copy tasks into them, but you can copy tasks from them. When you copy a task, any help associated with that task is not copied; the new task refers to the original help file. Use the right-click menu in Process Templates Setup to copy and paste tasks.

To add a task to a process

1. Select Edit menu > Process Templates.

   The Process Templates Setup dialog box appears.

2. Right-click a process in the left pane and select Add > Task.

   A new task appears below the process, with details at the right.

3. In Task Name, enter the name for this task.

4. In Type, define the type of task:

   - **Header**
     The task is an informational heading for a set of subtasks and does not require the user to do anything. Example: The header Package Testing is a heading for a group of subtasks.
     
     Header tasks appear in bold type and do not have check boxes or Run links.

   - **Manual**
     The task does not require the user to run a program or tool within Wise Package Studio. Example: It might require the user to review the results of the preceding task.
     
     Manual tasks do not have Run links.
Workbench Tool
The task requires the user to run a Workbench tool. On the Projects tab, clicking this task's Run link starts the tool.

See About Wise Package Studio tools on page 87.

When you select this option, the following fields appear:

♦ Tool
Select the tool this task will start. The list displays the predefined Workbench tools as well as any other tools you have added in Tool Setup.

♦ Options
(Optional.) Enter command-line options to change the default behavior of this tool or application.

See Guidelines for Entering Command Line Options on page 72 and Defining Command Line Options for Tools on page 73.

Other EXE
The task requires the user to run a program other than a Workbench tool. On the Projects tab, clicking this task's Run link runs the program's executable.

Example: You could set up this task to run reimaging software.

When you select this option, the following fields appear:

♦ EXE
Specify an executable on a local or network drive that this task should run. If you enter one that is on your local drive, other users who run this task must have the same executable stored in the same directory on their computers.

♦ Options
(Optional.) Enter command-line options to change the default behavior of this tool or application.

See Guidelines for Entering Command Line Options on page 72.

Pre-defined Application
This task requires the user to run a program other than a Workbench tool. When this task is displayed on the Projects tab, clicking its Run link starts the application. Example: If the purpose of this task is to document results, you can set up this task to run Microsoft Word.

When you select this option, the following fields appear:

♦ Application
This list contains all applications installed on your computer. Select the application that this task should run. Other users who run the task you create must have the same application installed on their computers, otherwise, the task will not run.

♦ Options
(Optional.) Enter command-line options to change the default behavior of this tool or application.

See Guidelines for Entering Command Line Options on page 72.

5. To prevent users from running tasks out of order, mark Verify that the file exists before the tool is launched. Do this if the file referenced in this task's command-line options must exist before the user can run the tool or program associated with this task. (The file name must be the last option in the command line.) Example: If
the previous task runs SetupCapture, and this task runs WiseScript Package Editor, you cannot run this task unless you’ve run SetupCapture to create the script file.

If the tool associated with this task creates the file, clear the check box. Example: Clear the check box if this task is SetupCapture, which creates an installation package.

This check box does not appear if this is a manual task.

6. If this tool requires an up-to-date, compiled installation program, mark Compile the package before the tool is launched. Example: If you are creating a task to distribute the final version of a Windows Installer package, you need to distribute the most current .MSI. When this check box is marked, the package installation is compiled before starting the tool if no compiled installation exists, or if the compiled file (.MSI, .MSM, or .EXE) is older than its associated project file (.WSI, .WSM, or .WSE, respectively).

7. (Enterprise Management Server only.) From Update project status upon task completion, select the status the project should have when this task is completed. Example: Suppose this task is “Resolve conflicts” and the next task is “Test package.” When the “Resolve conflicts” task is completed, you can change the project’s status to Testing.

If you select <Do not change>, the project’s status will not change when this task is completed.

This drop-down list does not appear if this is a header task.

8. (Optional.) Specify or enter help text to appear when a user clicks this task on the Projects tab. Mark one of the following options.

   See Help for Tasks and Tools on page 70.

   • HTML
     Mark this option if you have already created a help text file in a Web browser-compatible format such as .HTM or .ASP. When you mark this option, the Location field appears.

   • Rich Text
     Mark this option to create and edit the help text in .RTF format.

9. Save the new task by clicking Close or by clicking another task in Process Templates Setup.

Duplicating and Deleting a Process

➤ Not available in Standard Edition.

You can duplicate any process in Process Templates Setup. The predefined processes cannot be modified, so to use a variation of one of the predefined processes, you must duplicate it and modify the copy.

When you duplicate a process, any task help files associated with that process are not duplicated; the new tasks refer to the original help files. However, if you edit a help file within the new process, a new help file is created.

You also can delete processes from Process Templates Setup, but only processes that you have created, not predefined processes.

With Enterprise Management Server:
Security Setup determines whether you have access to Process Templates Setup.

You can duplicate processes from an external database, which places a copy in your local Workbench database, along with any user-defined Workbench tools or command-line options used in the process that are not already in your database. Processes that are copied from an external database are placed in the Local Processes tree in Process Templates Setup. Subsequent changes made to the corporate process template will not affect your local copy.

You cannot edit or delete an external process template.

**To duplicate a process**
1. Select Edit menu > Process Templates.
   The Process Templates Setup dialog box appears.
2. Right-click a process in the process list and select Duplicate.
   A copy of the process appears at the end of the process list.
3. Select the copy to modify it.
4. Save the new process by clicking Close or by clicking another process in Process Templates Setup.

**To delete a process**
1. Select Edit menu > Process Templates.
   The Process Templates Setup dialog box appears.
2. Right-click a process in the process list and select Delete.

Deleting a process template does not affect any existing projects, because each project contains its own a copy of the process. Deleting a process cannot be undone.

**Importing and Exporting Processes**
In Process Templates Setup, you can export a process template to a file (.WPR) and you can import a process template from a file. Example: You might create a process template in one Workbench database, export it, and then import it into another Workbench database. This lets you maintain a standard set of process templates across multiple distributed teams.

**Note**
With Enterprise Management Server, you can duplicate processes from an external database, which lets you more easily share standard process templates across databases.

See [External Process Templates](#) on page 60.

**To export a process**
1. Select Edit menu > Process Templates.
   The Process Templates Setup dialog box appears.
2. Right-click a process in the process list and select Export to File.
3. In the Save As dialog box that appears, specify a file name with the extension .WPR and click Save.
The .WPR file is created.

To import a process
1. Select Edit menu > Process Templates.
   The Process Templates Setup dialog box appears.
2. Right-click in the process list and select Import From File.
3. In the Open dialog box that appears, specify the .WPR file to import and click Open.
   The process appears in the process list.

Organizing Tasks and Processes

Not available in Standard Edition.

The arrangement of tasks in a process represents the order in which the tasks are typically performed. You can change the task order in Process Templates Setup. You also can change the order of processes in the process list, but this only affects the display; it does not affect the use of the processes. You cannot reorganize tasks in a predefined process.

Note
You cannot move a task outside its process. To put a task in a different process, copy and paste it using the right-click menu.

You can create a task hierarchy, with parent tasks and subtasks. To see an example of a task hierarchy, expand any of the predefined processes in the process list. When you create a task hierarchy, you might want to set up the first level of tasks as header tasks and use them to organize sets of subtasks.

See Adding Tasks to a Process on page 62.

With Enterprise Management Server, Security Setup determines whether you have access to Process Templates Setup.

To rearrange tasks or processes
1. Select Edit menu > Process Templates.
   The Process Templates Setup dialog box appears.
2. Click a task or process and click a move tool (↑ ↓ ← →).
3. Save the updated process by clicking Close or by clicking another task in the Process Templates Setup dialog box.

About Tool Setup

Not available in Standard Edition.

A tool is an executable application that you use to accomplish a task. In the Tools tab, you use a tool by double-clicking its icon. In the Projects tab, you use a tool by clicking the Run link to the right of the tool or the task that is associated with the tool.

Wise Package Studio contains predefined tools that should meet most of your needs.
Adding a New Tool

Not available in Standard Edition.

If you routinely use a third-party program in your processes, you can create a tool to run that program. This lets you use the tool in multiple tasks and processes without having to re-enter command-line options and other tool information each time. Example: If many of your processes require the user to use Microsoft Word, you can create a tool that runs Word.

With Enterprise Management Server, Security Setup determines whether you have access to Tool Setup. You also can use Security Setup to allow or restrict access to tools you create.

See Creating Groups and Setting Permissions on page 39.

To add a new tool
1. Select Edit menu > Tools.
   The Tool Setup dialog box appears.
2. Right-click in the dialog box and select Add.
   A new tool appears, with details at the right.
3. Specify the following tool properties:
   - **Name**
     Type the name for this tool.
   - **Tool Group**
     Select the group to which to assign this tool. This determines the location of the new tool on the Tools tab. This field is unavailable for predefined tools.
   - **Tool Type**
     Select the type of tool you are creating:
     - **Predefined Application**
       This tool runs an application, such as Microsoft Word or Notepad, that is installed on your computer.
       When you select this option, the Application list appears. This list contains most applications installed on your computer. Select the application this tool runs. Users who run this tool must have the same application installed on their computers, otherwise, the tool will not run.
Other EXE
This tool runs a program other than an installed application, or an application executable that is installed on a network drive. When you select this option, EXE and Icon appear. In EXE, specify the executable file on a network or local drive. You can specify a path relative to the location of Workbench.exe. If the executable is on the local drive, other users who run this task must have the same executable stored in the same directory on their computers. Icon shows the icon that is associated with this tool. Browse to select a new icon. If you change it, other users must have access to the new icon file.

Web Application
This tool runs a Web application in the right pane of Workbench.

Command Line
Enter command-line options to change the default behavior of this tool.

Hide from Tools tab in Workbench
Mark this check box to hide this tool in the Tools tab. When you hide a tool, you still can associate it with a task, but users cannot start it from the Tools tab. This prevents users from running the wrong tool accidentally and from using tools that should only be used from within a process. You also can use this check box to hide predefined tools that your company never uses. This option does not affect tasks associated with that tool or the display of tools in the Projects tab.

4. (Optional.) Specify or enter help text to appear when a user clicks this tool on the Tools tab. Mark one of the following options.

Adding a Web Application as a Tool
Not available in Standard Edition.

You can run a Web application from within Workbench by adding it as a tool.

To add a Web application as a tool
1. Add a new tool.
   Adding a New Tool on page 67.
2. From the Tool Type drop-down list, select Web Application.
Creating Projects, Processes, and Tools

3. **In URL**, enter the full path to the Web application, including the name of the computer on which the application resides. Example:
   
   http://computer_name/virtual_directory/application.asp

4. **Icon** shows the icon that is associated with this tool. Browse to select a new icon. Other users must have access to the new icon file you select.

5. Mark **Allow Local Override of URL** to let users specify a new URL in the Connection Failed dialog box that appears when Workbench cannot connect to the computer that is hosting this Web application. The new URL is stored in the user's registry instead of the Workbench database. If you clear this check box, then only users who have security access to Tool Setup can specify a new URL, because doing so changes the URL for all users.

6. When you change the URL for an existing tool, mark **Set URL for All Users** to save the new URL in the database and apply the change to all users who do not have a local override. If you clear this check box, the URL is changed in your registry and does not affect other users.

   If **Allow Local Override of URL** is cleared, this is marked by default and cannot be changed.

7. Complete the tool entry.
   
   See *Adding a New Tool* on page 67.

Duplicating, Deleting, and Rearranging Tools

➤ *Not available in Standard Edition.*

Use Tool Setup to duplicate, delete, or rearrange tools.

When you duplicate a tool, any help file associated with that tool is duplicated also.

With Enterprise Management Server, Security Setup determines whether you have access to Tool Setup.

**To duplicate a tool**

1. Select Edit menu > Tools.
   
   The Tool Setup dialog box appears.

2. Right-click a tool and select Duplicate.
   
   A copy of the tool appears at the end of the tool list.

   See *Adding a New Tool* on page 67.

3. Save the new tool by clicking Close or by clicking another tool in Tool Setup.

**To delete a tool**

**Warning**

Do not delete a tool that is associated with any tasks because it disables the task and prevents you and other users from running it.

1. Select Edit menu > Tools.
   
   The Tool Setup dialog box appears.

2. Right-click a tool and select Delete.
Deleting a tool cannot be undone. As an alternative, you can hide it from view in the Tools tab by marking its **Hide from Tools tab in Workbench** option.

**To rearrange tools**
The tools appear on the Tools tab in the same order they appear in Tool Setup.

1. Select Edit menu > Tools.
   The Tool Setup dialog box appears.
2. Select a tool and click ↑ or ↓.
3. Save the new tool order by clicking Close or by clicking another tool in Tool Setup.

This only affects the display; you can always use the tools in any order. This does not affect the display of tools on the Projects tab.

**Note**
No matter where you move a tool in Tool Setup, it will not be displayed outside its tool group unless you change the **Tool Group** field.

---

**Help for Tasks and Tools**

➤ *Not available in Standard Edition.*

The Description tab in the right pane of the Projects and Tools tabs displays help text on the currently selected task or tool. If you create a process, tool, or task, you can write help in HTML or .RTF format and associate the help file with the task or tool. Write HTML format help outside Workbench. You can write .RTF format help text directly in Workbench.

With Enterprise Management Server, Security Setup determines whether you have access to Process Templates Setup and Tool Setup.

**Using HTML for help**

In Process Templates Setup or Tool Setup, mark the **HTML** option and in the **Location** field that appears, enter one of the following:

- If the help file is located in the Workbench directory under your share point directory, enter the file name. You do not have to include the full path.
- If the help file is not located in the Workbench directory, enter the full path (Example: C:\Development\MyHelp.htm). To browse for the file, click Options and select Browse.
- Enter a URL (Example: www.Company.com/MyHelp.htm).
- If a help file is already specified, you can edit it; click Options and select Edit. The file opens in your default HTML editor or in Notepad.

**Using rich text for help**

In Process Templates Setup or Tool Setup, mark the **Rich Text** option to show a simple text editor for authoring in .RTF format. The help text you associate with a tool or task is saved in the Workbench directory under your share point directory.
Type directly in the help text editor to author help. The text editor provides common formatting tools you can use to format the help text. The help text editor also contains the following tools:

- **Insert Object**
  Inserts an object, such as an image, into your help text. Click this to open a standard Windows Insert Object dialog box, where you can select the object to insert.

  Available objects can vary depending on the computer. If you add an object that other users do not have on their computers, they will not see the object in the help text.

- **Edit with WordPad**
  If Microsoft WordPad is installed on your computer, click this tool to open the help text as a WordPad document. This lets you use WordPad formatting and other features to create and edit the text. When you save the file and close WordPad, the updated help text appears in the help text editor.

- **Edit with Word**
  If Microsoft Word is installed on your computer, click this tool to open the help text as a Word document. This lets you use more advanced Word features to create and edit the text. However, because the help file is saved in Rich Text Format, some of the formatting, such as tables, might not be saved. When you save the file and close Word, the updated help text appears in the help text editor.

**Note**
When you edit with either WordPad or Word, you cannot do anything else in Wise Package Studio until you save the text file and close WordPad or Word.

**See also:**
*Adding Wise Package Studio Variables to Help Text*

**Adding Wise Package Studio Variables to Help Text**

*Not available in Standard Edition.*

You can add a Wise Package Studio variable to tool or task help, whether the help is in HTML or .RTF format. When the help is displayed in Workbench, the value of the variable is displayed.

**Example:**
Suppose one of your processes contains a task to run SetupCapture and create a Windows Installer project file. To have the help text for that task display the file's location and name, add the following variables to the help text:

```
[ProjectDir]\[ProjectName].wsi
```
When the user displays the process in the Projects tab, the values of [ProjectDir] and [ProjectName] are displayed (Example, V:\Wise Share Point\Projects\Application1\Application1.wsi).

**Note**
When you display tool help in the Tools tab, the values for project-related variables are not displayed because the tools are not associated with projects.

See *Wise Package Studio Variables* on page 75.

## Command Line Options

*Not available in Standard Edition.*

You can use command-line options to change the way a tool or task runs in Workbench. Example: You can set an option that causes Windows Installer Editor to open the default project package automatically. Enter command-line options when you create a tool in Tool Setup or when you create a task in Process Templates Setup.

See *About Wise Package Studio command-line options* on page 268 and *Guidelines for Entering Command Line Options*.

Most predefined tools require a command-line option to run. In general, you should not change the command-line options of predefined tools, or the tool might not run properly.

To customize the way a predefined tool runs, it is best to add command-line options when you associate that tool with a task in Process Templates Setup. There, you can select from predefined command-line options for that tool. These predefined options have plain English descriptions that make it easier to understand what each command-line option does. You can define options for the predefined tools or for any tool you create.

See *Defining Command Line Options for Tools* on page 73.

**Warning**
Changing an existing command-line option affects all existing projects and processes that use that option.

## Guidelines for Entering Command Line Options

When you add a new tool or when you add tasks to a process, you can enter command-line options to apply to the tool that runs (optional). In Process Templates Setup, you add command-line options to the **Options** field. In Tool Setup, you add command-line options to the **Command Line** field.

See *About Command Line Options for Tools* on page 73.

Follow these guidelines when entering command-line options:

- If there is a right-arrow button next to **Options** or **Command Line**, you can click the button and select a Wise Package Studio variable.
  See *Wise Package Studio Variables* on page 75.
- If there is a Define button next to **Options** or **Command Line**, you can define a new command-line option.
Creating Projects, Processes, and Tools

See *Defining Command Line Options for Tools* on page 73. After defining a command-line option, it appears in **Options**.

- If you are working with a predefined Workbench tool, the **Options** field changes to a drop-down list and you can select a predefined command-line option. See *About Wise Package Studio command-line options* on page 268.

- If you are working with a tool that is not included in Workbench, see the program’s documentation for information about its command-line options. Example: If you select Word as the tool, you can enter a command-line option to pass the document name when Word is opened, so that Word opens the named document.

- If the command-line option contains a path or file name, enclose it in quotation marks.

### About Command Line Options for Tools

When you create a tool, you can enter command-line options that affect how the program runs. Command-line options you enter in Tool Setup affect the tool’s behavior every time the tool is run from the Tools tab, and in every task that uses that tool. Therefore, in Tool Setup, you should enter only the minimum command-line options that are required to run the tool. To produce more specific tool behavior on the task level, you can enter command-line options in Process Templates Setup, when you create a task that runs a tool.

Example:

Package Distribution requires the following option to run:

```
workbench.exe /tool="Package Distribution"
```

That is the option specified in Tool Setup. Adding the option `/tgt=23` to the command line runs Package Distribution and selects the option to distribute to a network directory. Because you will not want to distribute to a network every time you run Package Distribution, the `/tgt=23` option is added to a task in Process Templates Setup rather than to the tool in Tool Setup.

### Defining Command Line Options for Tools

*Not available in Standard Edition.*

See *Guidelines for Entering Command Line Options* on page 72.

**To define a command line for a tool**

1. Access the Tool Configuration dialog box in any of the following ways:
   - In Workbench, select Edit menu > Tool Configuration.
   - In Tool Setup or Process Templates Setup, right-click and select Tool Configuration.
   - In Process Templates Setup, select any tool, make sure **Task Type** is set to **Workbench Tool**, and click Define next to **Options**. When you access the dialog box this way, you can only edit options for the currently displayed tool.
     The Tool Configuration dialog box appears.
2. From Tool, select the tool to define options for. The list contains both predefined tools and tools you create.
3. Click Add.
   A new option appears at the end of Options.

4. Complete the lower section of the dialog box:
   - **Name**
     Type a description of the tool. (Example: If this command-line option opens a Word document, enter Open default document.)

   - **Command Line**
     (Optional.) Enter command-line options to change the default behavior of this tool or application.

   - **Auto Task Checking**
     Select a method for automatically completing tasks associated with this tool.
     - Do not mark task complete when tool finishes
       The task check box is not marked automatically.
     - Always mark task complete when tool finishes
       The task check box is marked the first time the tool closes—even if you cancel it—regardless of whether it ran successfully.
     - Mark task complete if tool finishes successfully
       The task check box is marked only when the tool is closed normally. If the tool has a wizard interface, the wizard must be completed through the final page. Canceling the tool does not mark the task complete.

   **Note**
   This option does not work for user-defined tools because it requires a return code from the application.

   - Mark task complete if following file changes
     The task check box is marked only when the file specified in the File Path field changes. Example: If you have a task that runs InstallTailor, you might want it to be marked complete only when the transform file is created.

     When you select this option, the File Path field appears. Enter the path to the file that must be changed or created in order for the task check box to be marked. Because this tool can be associated with tasks in many different projects, you should use Wise Package Studio variables. Example: If this tool is Microsoft Word, you might enter [ProjectDir]\[FileName].doc to represent the document that is created in the project directory with the project file name.

5. To move the new option to a different place in the options list, select the option and click Move Up or Move Down.

6. Click OK.
   The next time a user associates this tool with a task in Process Templates Setup, the new command-line option will be available.
Wise Package Studio Variables

In some areas of Wise Package Studio, you can use Wise Package Studio variables to represent files, directories, and other information. (Example: You can use the [Sharepoint] variable to represent the current share point directory.) Variables are used in:

- Command-line options for tools associated with an application or other .EXE in Process Templates Setup and Tool Setup.

- Help text (Rich Text or HTML) in Process Templates Setup and Tool Setup.
  See Help for Tasks and Tools on page 70.

- The Project Directory and Vendor Package fields in Project Setup.

The value for most variables is obtained from information in Project Setup. Some variables are not available in all Wise Package Studio editions and modules.

Available variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ApplicationName]</td>
<td>The current project’s Application Name field in Project Setup</td>
</tr>
<tr>
<td>[FileName]</td>
<td>The current project’s File Name field in Project Setup</td>
</tr>
<tr>
<td>[Notes]</td>
<td>The current project’s Notes field in Project Setup</td>
</tr>
<tr>
<td>[PackageName]</td>
<td>The current project’s Package Name field in Project Setup</td>
</tr>
<tr>
<td>[PackageStudioDir]</td>
<td>The directory in which Wise Package Studio is installed</td>
</tr>
<tr>
<td>[ProductVendor]</td>
<td>The current project’s Product Vendor field in Project Setup</td>
</tr>
<tr>
<td>[ProjectDir]</td>
<td>The current project’s Project Directory field in Project Setup</td>
</tr>
<tr>
<td>[ProjectName]</td>
<td>The current project’s Project Name field in Project Setup</td>
</tr>
<tr>
<td>[Sharepoint]</td>
<td>The current share point directory specified on the Workbench Preferences dialog box &gt; Repository tab</td>
</tr>
<tr>
<td>[SoftwareManagerDSN]</td>
<td>The Software Manager database that is associated with the current share point directory</td>
</tr>
<tr>
<td>[VendorPackage]</td>
<td>The current project’s Vendor Package field in Project Setup</td>
</tr>
<tr>
<td>[WorkbenchDSN]</td>
<td>The Workbench database that is associated with the current share point directory</td>
</tr>
</tbody>
</table>
Chapter 4
Repackaging Applications and Managing Projects

This chapter includes the following topics:

- About the Project and Tools tabs on page 76
- Using the Projects Tab on page 76
- Using the Tools Tab on page 78
- Connecting to a Web Application on page 79
- Managing Projects on page 80
- Viewing Project Metrics on page 83
- Creating a To-Do List on page 84
- Workbench Reports on page 85

About the Project and Tools tabs

To repackage or test applications in Wise Package Studio, you use either the Project tab or the Tools tab. Both tabs let you run the tools that create, modify, and verify packages.

With Enterprise Management Server, the right pane of the Projects tab contains additional tabs to help you manage projects. These include the Project Management, Metrics, and To Do tabs.

See:
Using the Projects Tab on page 76
Using the Tools Tab on page 78

Using the Projects Tab

Use the Projects tab to work on projects that you create in Project Setup. See About Projects on page 54.

When you create a project, you can associate it with a process. (See About Process Templates and Tasks on page 58.) If you associate a project with a process, you can take advantage of the process-oriented approach to repackaging applications in Wise Package Studio, which provides consistency in your repackaging and ensures that projects are performed according to corporate standards. If a project is associated with a process, the Projects tab displays the tasks that make up the process.

If you don't associate a project with a process, the Projects tab displays the Workbench tools, which you can use in any order. Because tools on the Project tab are associated with a project, you have greater control of the name and location of the job’s files than you do when working with tools on the Tools tab.
With Enterprise Management Server, Security Setup determines whether you have access to the Projects tab.

**To work on a process-oriented project**

*Not available in Standard Edition.*

1. Click the Projects tab or press Alt+P.

2. From **Active Project**, select a project that is associated with a process.

   A project is associated with a process when a process is selected in the **Process** field in Project Setup.

   The project name appears in **Active Project**, and its tasks appear in the left pane. Tasks that do not have check boxes are informational headers used to group subtasks. If a task is unavailable, it means you do not have a license to use the tool associated with it. With Enterprise Management Server, it might also mean you do not have permission to use the task’s tool or the task has not been assigned to you.

3. Starting with the first task that has a check box, do the following for each task, in order:

   a. Click the task, then click the Description tab to display the task’s help text. If the View Tool Help hotlink appears in the upper right of the Description tab, click it to access the tool’s help text.

   b. If a Run link appears to the right of a task, the task is associated with a tool. Click the Run link to run the tool.

   **Note**

   When you click the Run link, if the tool program cannot be found on your computer, a dialog box appears that lets you browse for the program. Wise Package Studio records the location you specify so it can find the program the next time you run that tool.

   When you run a tool from the Projects tab, the tool might skip dialog boxes or populate fields based on command-line options defined in Process Templates Setup. Example: The task Edit package might run Windows Installer Editor and open the project’s package file.

   c. If a task is not associated with a tool, do what the task describes. Example: Install Software.

   d. As you finish each task, mark the check box to the left of the task to indicate that the task is complete. Tasks that are associated with tools can be marked complete automatically, if the tool’s command-line option is set up to do so.

   See *Defining Command Line Options for Tools* on page 73.

4. When the entire project is finished, you can change its status.

   - In the Professional Edition, change the **Status** field in Project Setup to **Closed**.

   - With Enterprise Management Server, change the **Status** field on the Project Management tab to **Complete**, or set up the last task in the process to change the status automatically. You set task options in Process Template Setup.

   See *Adding Tasks to a Process* on page 62.
To work on a project without a process

1. Click the Projects tab or press Alt+P.
2. From **Active Project**, select a project that has no process.
   
   A project has no process when **None** is selected in the **Process** field in Project Setup.

   The project name appears in **Active Project**, and the tools appear in the left pane.
3. Click a tool name to display the tool’s help text.
4. To run a tool, click the Run link to the right of the tool name.
   
   When you run a tool from the Projects tab, you might not be prompted for file names or locations, because they usually are defined with the project.

To hide or show a tool window when it runs

- When a tool runs, the Run link changes to a Hide link. To minimize the tool window, click the Hide link.
- When a tool window is hidden, the link changes to Show. To restore the tool window to its last size and position, click the Show link.

For information about using specific tools, see:

*List of Wise Package Studio tools* on page 87
*About Capturing Applications* on page 201
*About ConflictManager* in the ConflictManager Help
*About Linux Package Editor* in the Linux Package Editor Help
*About Mobile Device Package Editor* in the Mobile Device Package Editor Help
*About Software Manager* in the Software Manager Help
*About Virtual Package Editor* in the Virtual Package Editor Help
*About Windows Installer Editor* in the Windows Installer Editor Help
*About WiseScript* in the WiseScript Package Editor Help

## Using the Tools Tab

The Tools tab displays all available tools and lets you use the tools in any order. You might use the Tools tab to perform a single task or a series of tasks without creating a project.

With Enterprise Management Server, Security Setup determines whether you have access to the Tools tab.

To run a tool from the Tools tab

1. Click the Tools tab or press Alt+T.
   
   The tool names and icons appear. Click a tool name to display the tool’s help text.
2. To run a tool, double-click the tool name.
   
   If a tool is not displayed or is unavailable, it means you do not have a license to use it. With Enterprise Management Server, it might also mean you do not have permission to use it.

For information about using specific tools, see:
Connecting to a Web Application

A Wise Package Studio task or tool can run a Web application whose URL is defined in Tool Setup. When Workbench cannot connect to the computer that is hosting the Web application, the Connection Failed dialog box appears.

- The **URL** field displays the path to the Web application. If this field is enabled, and if the Web application is on multiple servers, you can specify a new URL. Enter the full path to the Web application, including the name of the computer on which the application resides. Example:
  http://computer_name/virtual_directory/application.asp

Specifying a new URL might change the URL for other users. See below.

- If you do not specify a new URL, the Connection Failed dialog box might continue to appear as you use Workbench. To prevent future connection attempts and disable the dialog box, mark **Prevent Connection Attempts to Web Applications**.

  To activate the dialog box, select Edit menu > Preferences > General tab and mark the **Allow Connection Attempts to Web Applications** check box.

What happens when you specify a new URL?

The **Allow Local Override of URL** check box in Tool Setup determines what happens when you specify a new URL.

- If the check box is marked, the new URL is stored in your registry instead of the Workbench database. Therefore, it does not change the URL for other users.

- If the check box is cleared, the new URL is stored in the Workbench database and changes the URL for all users.

Why is the URL field unavailable?

- The **Allow Local Override of URL** check box in Tool Setup is cleared.

And

- You have a license for Enterprise Management Server and you do not have permission to access Tool Setup.

Because specifying a new URL without a local override changes the URL for all users, only users with permission to edit tools have this ability.
Managing Projects

 Enterprise Management Server only.

Use the Project Management tab in the right pane of the Projects tab to record information about a project at various stages of its lifecycle. You can then manage and track the project’s progress. You can also generate several Workbench reports that use this information.

To use the Project Management tab, see:

- Entering Project Tracking Information
- Assigning Users to Tasks in a Project on page 81
- Entering Time for Tasks on page 82

Entering Project Tracking Information

 Enterprise Management Server only.

Security Setup determines whether you have access to the Project Management tab.

Note
To edit a project’s tracking information, you must be the project owner, a user in the owner group, or a user in the WPS Administrator group. You cannot edit project tracking information for the predefined projects: Sample Project and Initial Workbench Setup. You can make a duplicate of these projects and edit them.

See Duplicating or Deleting a Project on page 57.

To enter project tracking information

1. Click the Projects tab or press Alt+P.
2. From Active Project, select a project that is associated with a process.
3. In the right pane, click the Project Management tab.
4. Complete the tab:

   ■ Project Owner
   This field displays the owner assigned in Project Setup. To change the owner, click ... and select an owner from the Select Owner dialog box. The owner can be a user, a group, or none. If you select a user, a dialog box appears asking you if you want to assign all the project’s task to that user.

   ■ Priority
   Select High, Medium, or Low.

   ■ Status
   Select a status if it does not get updated automatically.

   You can have a status updated automatically upon the completion of certain task.

   See Adding Tasks to a Process on page 62.

   A project can have any of nine statuses; use the ones that meet your corporate standards.
Note
You also can update the project status in Project Setup. When you change the status in Project Setup, it is changed in the Project Management tab as well, and vice versa.

- **Estimated Completion Date**
  Specify the date on which you expect the project to be completed.

- **Current Target Date**
  If you determine that the project will be completed before or after the estimated completion date, specify the new target date.

- **Actual Completion Date**
  This is pre-filled when Status is changed to Complete.

- **Estimated Hours**
  Enter the number of hours you estimate the project will take to complete.

- **Hours Completed**
  Enter the total number of hours that have been spent on the project. The **Enter actual hours by** field in Workbench Preferences determines how the hours are entered:
  - If Project is marked, you enter the total number of hours for the project, updating the entry as work on the project progresses.
  - If Task is marked, you enter the actual hours for individual tasks and the total of the task hours is entered here automatically.
    See Entering Time for Tasks on page 82.

- **Remaining Hours**
  After work on the project begins, use this field to record the number of hours of work that remain. Example: Suppose you originally estimated the project to take 40 hours, and you’ve completed 20 hours. However, because you know the remaining tasks will take 30 hours, you enter 30 hours here. This provides a more realistic number in the % Completed field.

- **% Completed**
  - If Remaining Hours is blank,
    \% Completed = Hours Completed / Estimated Hours
  - If Remaining Hours has a value,
    \% Completed = Hours Completed / (Hours Completed + Remaining Hours).

### Assigning Users to Tasks in a Project

➤ **Enterprise Management Server only.**

Repackaging jobs are often performed by a team. (Example: The project manager creates and assigns the project. An integrator might analyze and customize the existing package. A repacker might perform the integration testing, and a tester might verify the package and finish the project.) On the Project Management tab, you can assign users to specific tasks in a project.

All tasks in the predefined projects are assigned to the user Admin. All tasks in a new project are initially assigned to the user who created the project.

Security Setup determines whether you have access to the Project Management tab.
To assign users to tasks in a project
1. Click the Projects tab or press Alt+P.
2. From Active Project, select a project that is associated with a process.
3. In the right pane, click the Project Management tab.
   The tasks for the selected project appear.
4. Select one or more tasks to assign.
5. Click Assign User and select a user from the button menu.

Only the tasks assigned to the current user are enabled. Tasks assigned to other users are visible but unavailable.

You can filter the projects so that only projects whose next task is assigned to you or only projects that have any task assigned to you appear in Active Project. Options for the project filter are set on the General Preferences dialog box.

See Setting General Preferences on page 52.

You can run the Assignments by User report to see how many tasks are assigned to each user.

See Workbench Reports on page 85.

Entering Time for Tasks

> Enterprise Management Server only.

On the Project Management tab, you can enter the time spent on tasks. This provides data for determining the difference between estimated and actual time spent on project tasks. Having historical data will help improve the accuracy of estimates for future repackaging projects. You can view this information in the Project Variance report.

See Workbench Reports on page 85.

Security Setup determines whether you have access to the Project Management tab.

To enter time for a task
1. Click the Projects tab or press Alt+P.
2. From Active Project, select a project that is associated with a process.
3. In the right pane, click the Project Management tab.
   The tasks for the selected project appear.
4. Double-click a task.
   The Task Details dialog box appears.
5. In Actual Hours, enter the time spent on this task. Up to two decimal places are allowed.

Don’t use the value from the Measured Hours column that gets recorded automatically. This represents only the time it takes to run the tool for a task, but a task might involve activities in addition to running the tool. Example: Suppose you need to test the package’s launch conditions on different operating systems. The Metrics tab will record the amount of time the Test Expert tool runs, but it will not record the time you spend reimaging the test machines.
6. Click OK.

The hours appear in the Actual Hours column. If Enter actual hours by in Workbench Preferences is set to Task, the sum of the task hours appears in Hours Completed.

**Viewing Project Metrics**

➤ Enterprise Management Server only.

The Metrics tab in the right pane of the Projects tab displays a record of all events that have occurred for the current project. This includes how much of the project has been completed, how many tasks remain, the time spent on the project to date, the sequence in which tasks were performed, who worked on each task, and how long each task took to complete. Because the Metrics tab records information each time a task is run, you can investigate false starts and task failures. When a project is completed, you can use its metrics information to help you estimate future projects.

The Metrics tab is available only for projects that have processes. Security Setup determines whether you have access to the Metrics tab.

**Project-level information**

The following information is displayed at the top of the Metrics tab:

- **Actual Start Date**
  If the first task is associated with a tool, the start date is recorded when a user first clicks the Run link for that task. If the first task is manual, the start date is recorded when the user marks the task’s check box or moves to the next task.

- **Actual Completion Date**
  The date when the last task in the project is marked complete.

- **Elapsed Time**
  The total amount of time (in HH:MM:SS format) the project has been open in the Active Project list. This will not match the Total Measured Time because it does not measure the time for manual tasks, or a user might have the project open without working on a specific task.

- **% of Tasks Completed**
  The number of tasks completed divided by the total number of tasks.

- **# of Tasks Remaining**
  The number of tasks that are not complete.

- **Total Measured Time**
  The total of the measured times for all tasks, in HH:MM:SS format.

**Task-level information**

The list section of the Metrics tab displays an entry for each task in the project. Information for a manual task entry is recorded when the task’s check box is marked or cleared, or when the user moves to the next task. Information for a tool-related task entry is recorded when the task’s Run link is clicked, or when the task’s check box is marked or cleared.

Each time a task that is associated with a tool is run, a tool entry appears as a child of the task entry. To display tool entries, click the plus sign to the left of a task entry. To
expand or collapse all the tool entries, right-click anywhere in the list and select Expand All or Collapse All.

You cannot edit or delete the task or tool entries, but you can enter notes for each task entry. To view or edit notes, click the link in the Notes column.

**Interpreting a task’s status**
- If a task’s tool is run and finished, or if its check box is marked, the task’s Result is Completed.
- If a task’s check box is cleared, the Result is blank for the task entry and Failed for the tool entry.

**Creating a To-Do List**

*Enterprise Management Server only.*

The To Do tab in the right pane of the Projects tab displays to-do items for the current project. Members of the repackaging team create the to-do items and can assign them a user and due date. When an item’s due date has passed, it is displayed in red.

To-do items differ from tasks in that they are unique to a project. They might represent actions that don’t warrant being entered as a task, or issues that arise during the project. (Example: Suppose the first task in your project is Analyze vendor package and a later task is Test package. The person who performs the analysis task can make notes about issues discovered during the analysis, and enter to-do items to record specific things the tester needs to do during the testing task.) In a project that has no process, you can use the to-do list to provide general guidelines for the users who will be working on the project.

Information on the To Do tab is used to generate the To Do List report.

See *Workbench Reports* on page 85.

Security Setup determines whether you have access to the To Do tab.

**To create a to-do list**

1. Click the Projects tab or press Alt+P.
2. From *Active Project*, select a project.
3. Click the To Do tab.
4. On the first line of the list, click **Click here to add a new item** and type the to-do item.
5. To assign the item to a user, click the **User Name** column and select a user.
6. To assign a due date to the item, click the **Due Date** column and select a date.
7. When you finish entering the to-do information, click outside the to-do item.
   - The item is added to the end of the to-do list.
   - To sort the to-do list, click any column heading.
   - To delete a to-do item, right-click the item and select Delete.
   - To mark a to-do item as complete, mark the check box to the left of the item.
Workbench Reports

*Not available in Standard Edition.*

Workbench contains reports that provide information about Workbench processes and projects.

In Enterprise Management Server, the Management Reports Web application lets managers of repackaging teams generate predefined Workbench reports without purchasing additional Wise Package Studio licenses.

## Workbench Reports

<table>
<thead>
<tr>
<th>Report Name</th>
<th>Available in</th>
<th>What This Report Does</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process Documentation</td>
<td>Professional Edition</td>
<td>Provides an overview of all Workbench processes by listing the tasks, tools, and options in each process. Use this report to document and compare Workbench processes. Not available from the Management Reports Web Application.</td>
</tr>
<tr>
<td>Project Overview</td>
<td>Enterprise Management Server</td>
<td>Lists general information about all Workbench projects. A repackaging team leader can use this report to get an overview of current projects.</td>
</tr>
<tr>
<td>Project Details</td>
<td>Enterprise Management Server</td>
<td>Provides information about specific Workbench projects. It contains the same information as the Project Overview report but also lists each project's tasks, the user assigned to each task, and each task's status. A project leader can use this report to check the status of individual tasks for a project.</td>
</tr>
<tr>
<td>Project Variances</td>
<td>Enterprise Management Server</td>
<td>Lists project estimated and actual completion dates and times and their variances. You can generate this report for all projects or projects with a particular status. Use this report to help improve the accuracy of estimates for future projects.</td>
</tr>
<tr>
<td>Assignments by User</td>
<td>Enterprise Management Server</td>
<td>Lists each user's assigned projects with the number of their assigned and completed tasks for each project. Repackagers can use this report to identify the projects assigned to them. A repackaging team leader can use this report to assess the workload balance among the members of the repackaging team.</td>
</tr>
<tr>
<td>To Do List</td>
<td>Enterprise Management Server</td>
<td>Lists all items on the To Do list for each project. Not available from the Management Reports Web Application.</td>
</tr>
<tr>
<td>Security Setup</td>
<td>Enterprise Management Server</td>
<td>Lists by group the user name and domain names of those with Wise Package Studio user permissions. The person responsible for Workbench security can use this report to get a list of the users assigned to each security group.</td>
</tr>
</tbody>
</table>
Generating a Workbench Report

To generate a Workbench report

➤ Not available in Standard Edition.

1. From the Reports menu, select a report.
   The report opens in the report viewer window.

2. From the report viewer window, you can do the following:
   ■ View the date and time when the report was generated in the lower left of the report.
   ■ Save the report by clicking Save As in the lower right of the report viewer window. You can save a report in HTML, XML, or CSV format.
   ■ Print the report by clicking Print in the lower right of the report viewer window.
   ■ Perform a text search by clicking within the report and typing Ctrl+F.

To generate a report through the Management Reports Web application

➤ Enterprise Management Server only.

1. In your browser, enter the URL for Management Reports.
   The Management Reports URL is http://server name/Wise_Management_Reports, where server name is the name of the IIS server. Obtain the server name from your administrator.

2. From Select a report to generate, select a report.
   The report opens in the Web report viewer window.

If you have problems using the Management Reports, ensure that you have met the system requirements for Web applications that are listed in the Wise Package Studio Getting Started Guide.
Chapter 5
Wise Package Studio Tools

This chapter includes the following topics:

- About Wise Package Studio tools on page 87
- List of Wise Package Studio tools on page 87
- How Wise Package Studio tools interact with revision control on page 89
- Application Isolation on page 89
- ApplicationWatch on page 94
- Command Line Builder on page 97
- InstallTailor on page 105
- Legacy Setup Conversion on page 110
- Package Definition on page 123
- Patch Creation on page 128
- UpgradeSync on page 138
- Web Capture Conversion on page 141
- Wise Task Manager on page 141
- Adding Files From the Wise Software Repository on page 144

About Wise Package Studio tools

Run Wise Package Studio tools from the Tools tab or the Projects tab.

- On the Tools tab, the tools are organized to correspond to the phases of the application management lifecycle. To run a tool, double-click the tool name.
- On the Projects tab, you click the Run link to the right of the tool or, in a project with a process, to the right of the task associated with the tool.

When you run a tool from the Projects tab, the tool might skip pages or populate fields based on command-line options defined in Process Templates Setup. Example: The task Edit package might run Windows Installer Editor and open the project’s package file.

Note
When you specify a .WSI or .WSE file in certain tools, the file is compiled if necessary. If any files cannot be read, or if other errors are encountered during the compile, a dialog box listing the errors appears. Open the package in its editor (Windows Installer Editor or WiseScript Package Editor), fix the errors, and rerun the tool.

List of Wise Package Studio tools

- Application Isolation. (Not available in Standard Edition.)
- **ApplicationWatch.**
- **Command Line Builder.**
- **ConflictManager®,** which helps you solve the problem of conflicting files and registry entries that often occur on end user computers, letting you avoid problems when deploying packages throughout your organization. See *About ConflictManager* in the ConflictManager Help. (Not available in Standard Edition.)
- **Impact and Risk Assessment,** which lets you quickly assess the potential impact of deploying a package (usually a hotfix or security patch) without performing extensive testing. It also lets you determine which isolated files are at risk of being missed by an update or patch and ensure that they are updated. See *Impact and Risk Assessment* in the Software Manager Help. (Not available in Standard Edition.)
- **InstallTailor.**
- **Legacy Setup Conversion.**
- **Linux Package Editor,** which lets you use a Windows computer to create packages that install software on Linux computers. See *About Linux Package Editor* in the Linux Package Editor Help.
- **Mobile Device Package Editor,** which lets you create an .INF file and compile it to one or more .CAB files that install a mobile device application. See *About Mobile Device Package Editor* in the Mobile Device Package Editor Help.
- **Package Definition.**
- **Package Distribution.**
- **About Package Validation.**
- **Patch Creation.**
- **Preflight Analysis.** See *Viewing Results from Preflight Deployment* on page 263.
- **Preflight Instrumentation.** See *Creating a Preflight Package* on page 262.
- **SOE Snapshot** on page 242. (Not available in Standard Edition.)
- **SetupCapture.**
- **SetupCapture Configuration.**
- **Software Manager,** which provides the interface for working with packages in the Software Manager database. See *About Software Manager* in the Software Manager Help. (Not available in Standard Edition.)
- **About Test Expert.** (Quality Assurance module only.)
- **UpgradeSync.**
- **Virtual Package Editor,** which lets you create and edit a virtual software layer, a virtual software project (.WVP) file, or a virtual software archive (.VSA) file. See *About Virtual Package Editor* in the Virtual Package Editor Help. (Not available in Standard Edition.)
- **Web Capture Conversion.** (Not available in Standard Edition.)
- **Windows Installer Editor,** which is a development system for creating and editing Windows Installer installation packages (.WSI, .MSI). See *About Windows Installer Editor* in the Windows Installer Editor Help.
- **WiseScript Editor,** which is a script authoring environment that lets you create powerful .EXEs to use as custom actions in a Windows Installer installation. In
addition to being available in Workbench, WiseScript Editor is embedded within Windows Installer Editor and appears when you create a custom action that calls a WiseScript.

WiseScript Editor shares documentation with WiseScript Package Editor. See About WiseScript in the WiseScript Package Editor Help.

- WiseScript Package Editor, which is a development system for creating and editing installation packages based on the Wise scripting language (WiseScript™). See About WiseScript in the WiseScript Package Editor Help. (Not available in Standard Edition.)
- Wise Task Manager.

The MSI to WSI Conversion tool and Import Visual Basic, C#, or J# tools are available from the Tools menu in Windows Installer Editor. See About Windows Installer Editor tools in the Windows Installer Editor Help.

### How Wise Package Studio tools interact with revision control

When you specify a file that is under revision control in Software Manager, the tool you are using checks the file's revision control status. This table shows what the tool does for each revision control status.

<table>
<thead>
<tr>
<th>If the file is</th>
<th>Wise Package Studio tool does this</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not under revision control</td>
<td>Opens the file</td>
</tr>
<tr>
<td>Checked in</td>
<td>Asks if you want to check out the file</td>
</tr>
<tr>
<td></td>
<td>If you click Yes, the tool checks out the file and opens it.</td>
</tr>
<tr>
<td>Checked out by you</td>
<td>Opens the file</td>
</tr>
<tr>
<td>Checked out by another user</td>
<td>Displays message saying the file is checked out by another user and is unavailable</td>
</tr>
</tbody>
</table>

You can view a package's revision control status in the Application/Package Summary pane in Software Manager.

See Revision Control in the Software Manager Help.

### Application Isolation

*Not available in Standard Edition.*

Application Isolation™ provides a quick and easy way to isolate applications with their shared .DLL or .OCX files (support files). It isolates the .EXE files in an installation by placing their dependent, shared .DLLs and .OCXs inside the application directory or, optionally, in the WinSxS directory.

Application Isolation ensures that an application always uses the version of shared files with which it was installed. It prevents overwriting of previous versions of shared files.
components, and ensures that other applications do not overwrite your version of shared components. This lets you proactively eliminate potential conflicts with other applications.

Application Isolation operates on Windows Installer installation files and transform files. You can save the output of Application Isolation as an .MSI or, to avoid violating a license agreement by changing the .MSI, you can save the output as an .MST.

**Warning**
Isolation does not work on all applications. Applications must be written according to Microsoft programming guidelines. (Example: If an application contains hard-coded paths to support files, isolation might not work.) Because ApplicationWatch records the support files accessed by an installation, use it to determine if the application follows Microsoft programming guidelines for accessing files.

See [ApplicationWatch](#) on page 94.

If you cannot use Application Isolation, use ConflictManager instead, which resolves conflicts rather than isolates applications.

Because Application Isolation has the potential to change the location of files within your installation and to change the feature and component layout of the installation database, test the package thoroughly after using Application Isolation.

**For further reading**
See [Isolated Components](#) in the Windows Installer SDK Help.

In the MSDN Library ([msdn.microsoft.com/library](http://msdn.microsoft.com/library)), search for the following terms:

- Assembly manifest
- Isolated components
- Isolated applications

### Creating a Package That Isolates .EXEs

➤ *Not available in Standard Edition.*

Create a package that isolates .EXEs to isolate applications with their shared .DLL or .OCX files (support files). You can save the results of the application isolation as an .MSI or an .MST.

Make sure your .MSI is complete or nearly complete so all files that need to be isolated are present in the package.

**To create a package that isolates .EXEs**

1. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with Application Isolation. Isolation will be run on the default project file. This tool might skip pages or populate fields based on command-line options defined in Process Templates Setup.
   - On the Tools tab, double-click Application Isolation.

2. If the Specify Target Installation File page appears, complete it:
- **Installation Pathname**
  Specify an installation in which to isolate .EXE files and click Next. You can specify an .MSI, an .MST, an .MSM, a .WSI, or a .WSM.

  If you specify an .MST, you can choose the same .MST in **Updated Windows Installer File** on the Isolation Complete page to append the results of the application isolation to that .MST, rather than overwriting it.

- **Base MST file**
  Specify a transform to apply to the installation.

3. On the Welcome page, mark your isolation method:

- **Use manifests to isolate application files**
  Adds .MANIFEST files to your installation. Manifests, a .NET technology, use metadata to describe in detail the relationships between an .EXE and its .DLL and .OCX files. You can specify that some support files are accessed from the application directory and some are not. This method of isolation only works when the installation is installed on a computer using Windows XP or later.

- **Use Windows Installer Isolated Components to isolate application files**
  This method of isolation offers less flexibility than manifests, but runs on more operating systems. Support files are placed in the application directory along with a .LOCAL file, which informs the operating system to access the .EXE's support files from the application directory.

4. On the Welcome page, mark your isolation type:

- **Automatically isolate application files**
  This option automatically isolates all applications with all support files in the installation. Example: Suppose your installation contains Sample1.EXE, Sample2.EXE, and Sample3.EXE, and also contains four .DLLs stored in the System32 directory. All four .DLLs will be copied to the application directory, and if any of the three .EXEs call any of the .DLLs, the system will direct the .EXEs to the isolated copies of the .DLLs in the application directory instead of the ones in the System32 directory.

- **Manually choose which files to isolate with which applications**
  If you mark this, an additional page appears that lets you manually choose which support files to isolate with which .EXE files. Example: If you isolate Sample1.EXE with a specific version of the file comctrl32.DLL, then a copy of comctrl32.dll will be isolated in the application directory along with Sample1.EXE. When Sample1.EXE calls comctrl32.dll, the system will direct it to the isolated copy.

5. On the Welcome page, click Next.

6. If you chose manifests, the Select OS Compatibility page appears.
   a. Mark options for Operating System Support and Side-by-Side Assembly Type.
   b. To have a copy of the isolated files installed in their original location, mark that option.
   c. Specify the extracted files directory.

   See **Specifying OS Compatibility for Isolation** on page 93.

7. If you chose Windows Installer isolated components, the Select Isolation Options page appears.
   a. Mark whether files should be moved to enable more comprehensive isolation.
b. Mark whether to add support for self-repair of isolated files.

See *Selecting Isolation Options* on page 94.

8. Click Next in either the Select OS Compatibility or Select Isolation Options page.

9. If you specified the option to manually select which files to isolate, the Select Files to Isolate page appears.

   a. From **Select Files to Isolate From**, select what kind of installation files should appear in the right list box.

      Typically, you only isolate .DLLs and .OCXs that are in shared locations, such as the System32 directory. However, to isolate other files, select **All files in shared locations** to view any type of file in a shared directory or **All files** to view all files in the installation.

   b. From **Application(s) to be Isolated**, select one or more .EXE files.

   c. In **Files to Isolate for Selected Applications(s)**, mark the files that should be isolated with the selected .EXEs.

      **Note**

      If, on the Select Isolation Options page, you chose not to move files to different features, then when you select an .EXE, you see only files in the same feature as the .EXE.

   d. Click Next when finished specifying isolation relationships.

10. If you chose to use manifests for isolation and to store files in the WinSxS directory, the Digital Signature Information page appears. You must digitally sign files that are stored in the WinSxS directory. Enter paths to the digital signing files you obtained from your digital signature provider such as Verisign and click Next.

11. On the Perform Isolation page, click Next.

    Isolation is performed, which might take a few minutes. Then the Isolation Report page appears, showing you the changes made to the installation to implement isolation.


    The Isolation Complete page appears.

13. In **Updated Windows Installer File**, specify whether to save the updated Windows Installer file as an .MSI or an .MST. You can also change the default name, which is the original file name with “_Isolated” appended.

    If you specified an .MST in **Installation Pathname** on the Specify Target Installation File page, you can specify the same .MST in **Updated Windows Installer File** to append the results of the application isolation to the .MST. If you specify an existing .MST in Updated Windows Installer file, but not the same .MST you specified in **Installation Pathname**, the results of the application isolation will overwrite the existing .MST.

14. Click Finish.

After you isolate an installation, do not add .EXE, .DLL, or .OCX files. The addition of files might nullify the isolation you performed.
Specifying OS Compatibility for Isolation

> Not available in Standard Edition.

In Application Isolation, the Select OS Compatibility page appears only if you chose to use manifests as the Isolation Method. Complete the page as follows:

To specify OS compatibility for isolation
1. Mark the operating system(s) to support:
   - **Support prior operating systems also**
     Because manifests are a .NET technology that work only on Windows XP or later, an installation using manifests will not run on any other operating system. This option lets you work around this limitation by configuring the installation so that it is compatible with any operating system. It creates an installation file with two copies of your installation. One is run if the operating system is Windows XP or later with isolation taking place using manifests. The other is run on all other operating systems with no isolation. This option significantly increases the size of the installation.
   - **Support Windows XP or later only**
     This option creates an installation that runs only on Windows XP or later and uses manifests for isolation.

2. Mark the assembly type:
   - **Create private side-by-side assemblies in application directory**
     Files and manifests, which together form assemblies, are installed to the application directory if they are isolated.
   - **Create shared side-by-side assemblies in WinSxS directory**
     Isolated assemblies are stored in the WinSxS directory, a global directory in the Windows directory for the side-by-side isolation of files. It provides a shared location, yet manages the isolation of files by comparing the digital signatures of files to determine differences. Files in the WinSxS directory must be digitally signed.

3. To also install the isolated files in their original location, such as System32, mark **Place copy of isolated files in their original location for application not written to support isolation**.
   This prevents the application from breaking in the event that the application you are isolating does not support isolation. Example: If it uses hard coded paths to access support files.

4. From **Extracted Files Directory**, select a directory for the isolated files.
   Isolated files are extracted and saved in this directory. When the installation is subsequently saved and compiled, these files are pulled from the new directory, not the original directory.
Selecting Isolation Options

Not available in Standard Edition.

To select isolation options

In Application Isolation, the Select Isolation Options page appears only if you chose to use Windows Installer Isolated Components as the Isolation Method. Complete the page as follows:

1. From Feature Options, mark one of the following options:
   - Move files into the same feature as necessary and then isolate
     When isolation is performed, files are moved up to parent features as needed until the .EXE and its support files are located in the same feature. Example: With automatic isolation, .EXEs and their support files are moved up in the hierarchy until they all reside in the same feature. With manual isolation, you can isolate any .EXE in your installation with any support file, and when isolation is performed, the files are moved as needed.
   - Isolate only those files that are already in the same feature as the application .EXE
     When isolation is performed, only .EXEs and their support files that reside in the same feature are isolated with each other. Example: Suppose Sample.EXE and Sample.DLL are in the same feature and you chose automatic isolation of files. When isolation is performed, Sample.EXE and Sample.DLL are isolated with each other. But if they were in different features, they would not be isolated with each other. If you chose manual isolation, only files in the same feature as the .EXE appear on the Select Files to Isolate page.

2. From Repair Support for Isolated Files, mark one of the following options:
   - Do not add repair support for isolated files
     Isolated files in the installation do not have self-repair.
   - Install isolated files in their original location
     Isolated files are installed in the application directory and in their original location, such as System32. The installation adds a component that facilitates repair of the files in the application directory. When the application needs to be repaired, Windows Installer looks for the files in both the system directory and the application directory.
   - Install isolated files in the application directory only
     Isolated files are installed in the application directory but not in the system directory. When the application is started, Windows Installer looks for the file in the application directory.

     If copies of the isolated file are also installed in other directories, then the installation creates component ID registry entries to refer to the other locations of the file. When the application needs to be repaired, Windows Installer looks for the file in the application directory and in any other directories referenced in the registry.

ApplicationWatch

ApplicationWatch™ monitors your computer as you execute an application or run an installation and determines which .DLL, .OCX, and .EXE files were accessed. It then adds
these files to a new installation. You can use this tool for informational purposes or to facilitate the creation of a new installation.

ApplicationWatch produces a Windows Installer or WiseScript package. (WiseScript is not available in Standard Edition.)

To completely recreate an installation, if you have the setup program that installed the application, use SetupCapture instead of ApplicationWatch. SetupCapture produces a complete record of the files and system changes made during installation, while ApplicationWatch records only the .DLL, .OCX, or .EXE files that are accessed during execution or installation of an application.

Note
ApplicationWatch cannot monitor 16-bit applications.

ApplicationWatch Exclusion List

Some files, such as system .DLL files, that are used during execution of an application are unrelated to the actual application. Because such files should not be added to the resulting package, ApplicationWatch™ uses an exclusion list to determine which files to ignore.

When you create a Windows Installer-based installation (.WSI or .MSI) with ApplicationWatch, a built-in exclusion list is used that cannot be changed. It contains operating-system related files that are accessed during the normal course of operating a computer and that typically have nothing to do with a specific application.

Likewise, when you create a WiseScript installation (.WSE) with ApplicationWatch (Professional Edition only), a built-in exclusion list is used. Though you cannot access this built-in list, you can add exclusions to it. To do this, open WiseScript Package Editor and select Edit menu > Preferences. In Preferences, enter the .DLLs to ignore in the System .DLLs to Exclude list.

Creating a Package with ApplicationWatch

ApplicationWatch produces a Windows Installer or WiseScript package. (WiseScript is not available in Standard Edition.)

(Professional Edition only.) When you work in a project with no process, the Projects tab displays two ApplicationWatch tools: ApplicationWatch (Windows Installer) that makes the target file a Windows Installer package and ApplicationWatch (WiseScript) that makes the target file a WiseScript package.

Note
ApplicationWatch cannot monitor 16-bit applications.

To watch an application

1. Close all other applications so that the files they access are not added to the installation.

2. Do one of the following:

   - On the Projects tab, click the Run link to the right of the task or tool associated with ApplicationWatch. The package that is created will be saved with the default project name. This tool might skip pages or populate fields based on command-line options defined in Process Templates Setup.
On the Tools tab, double-click ApplicationWatch.

3. If the Specify Target Installation File page appears, complete the page and click Next:

   **Target Installation**
   Specify the full path of a new or existing .MSI or .WSI file. In the Professional edition, you also can specify a .WSE file. The ApplicationWatch results will be stored in this file.

   **Add/Update Resources in Existing Installation**
   If you specified an existing installation in **Target Installation**, mark this to add to or update the resources in the existing installation instead of overwriting them.

4. Complete the Run Application page:

   **Application Path**
   Specify the full path of the application executable to run.

   **Command Options**
   Enter any command-line options to apply to the executable. Refer to the source application's documentation for applicable command-line options.

5. Click Execute, which starts the source application.

6. In the source application, use all possible features except printing. If you are watching an installation, run the installation through to completion.

   Use as many of the application's features as possible to ensure that files used by rarely-used features are recorded. Do not use the application to print, because printing accesses Windows operating system and printer-specific files.

7. Close the application, return to the Run Application page, and click Next.

   - In the Standard Edition, the Finish page appears. Click Finish and skip the next steps.
   - In the Professional Edition, the ApplicationWatch Inclusions page appears. It displays all files that were used during execution of the watched application. These files will be added to the new package file that is created, if you do not choose to remove them.

8. To exclude a file from the package, select it and click Exclude.

9. Click Next on the ApplicationWatch Inclusions page.

10. The ApplicationWatch Exclusions page appears. It displays all the files that are excluded from the new package. This includes files you excluded on the Inclusions page and files in the exclusion list.

    See **ApplicationWatch Exclusion List** on page 95.

11. To include a file in the new installation that is currently excluded, select it and click Include.

12. Click Finish on the ApplicationWatch Exclusions page.

    If a file that is part of a merge module is added, the Files in Merge Modules dialog box appears. It prompts you to add the merge module and, if necessary, download it.

    See **Adding Merge Modules Instead of Files** on page 232.
If a file that is used by a package in the Wise Software Repository is added, the Files in Repository dialog box appears and prompts you to add a version of the file that is in the repository.

See Adding Files From the Wise Software Repository on page 144.

13. If you created a .WSI or .MSI, open the new installation in Windows Installer Editor. If you created a .WSE file, open it in WiseScript Package Editor.

The files that were accessed by the source application are added to the Files page. If you use this information as a starting point for developing a complete installation, compile and test the installation thoroughly to verify that it operates correctly.

Warning
Some of the files that are listed on the Files page might be platform-specific or non-distributable Windows system files. If you are not sure whether it is safe to deploy a file, check with Microsoft developer documentation before deploying these files to end users.

Command Line Builder

Command Line Builder helps you create command lines for starting or updating Windows Installer packages without an extensive knowledge of command-line switches. You select the appropriate options for the package, and the utility creates an error-free command line for an optimal installation or update.

Command lines let you change the behavior of a package to account for different work environments and end user requirements. You can specify the following:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UI Options</td>
<td>Windows Installer provides package developers the ability to create user interfaces with multiple levels of functionality. Using a command line, you can control what level of functionality the end user sees during the installation.</td>
</tr>
<tr>
<td>Logging Options</td>
<td>Windows Installer records errors and events in log files. The diagnostic information Windows Installer writes to these logs can help you understand the cause of a failed installation. You can build a command line to specify the logging options you require for each package.</td>
</tr>
<tr>
<td>Advertising Options</td>
<td>Windows Installer can advertise the availability of an application to end users and other applications without actually installing the application. If an application is advertised, only the interfaces required for loading and starting the application are presented to the end user or other applications. If an end user or an application activates the advertised interface, Windows Installer loads the necessary components. This option can save time and disk space.</td>
</tr>
<tr>
<td>Repair Options</td>
<td>Windows Installer uses the REINSTALL property to repair installed components. Command-line options vary from reinstalling a minimum set of files to reinstalling all files, including registry entries.</td>
</tr>
<tr>
<td>Public Properties</td>
<td>Windows Installer changes the value of public properties to address specific installation requirements.</td>
</tr>
</tbody>
</table>
Creating a Command Line With the Command Line Builder

Use the Command Line Builder to create a command line to start or update a Windows Installer package.

For a description of the types of command-line options you can build, see Command Line Builder on page 97.

To create a command line

1. Start the Command Line Builder in either of two ways:
   - On the Projects tab, click the Run link to the right of the task or tool associated with Command Line Builder.
   - On the Tools tab, double-click Command Line Builder.
   The Welcome page appears.

2. In the Select File section, select the type of file to run with the command line:
   - MSI Package (*.msi)
   - MSI Package wrapped in a Wise .EXE (*.exe)

3. In the File Location section, specify the file to be run with the command line. To run a patch (.MSP) without an installation, leave this field blank.

4. Click Next.
   The Define Command Line page appears.

   **Note**
   Although you can enter a command-line in the Command Line field, we recommend that you use the options provided in this utility for an optimal, error-free installation.

5. Select an Install Mode. These options, with the exception of Update, are available only if you specified a file in the File Location section on the Welcome page:
   - **Install**
     Installs or configures the installation package. Select this option to remove patches.
   - **Advertised**
     Advertises the installation to the destination computer.
- **Repair**
  Repairs an application that is installed on the destination computer.

- **Network Install**
  Extracts the files in the installation package to a network location.

- **Uninstall**
  Uninstalls the installation package.

- **Update**
  Updates the installation package by applying patches. This is only available if you left the **File Location** section on the Welcome page blank.

6. Depending on the install mode you select, additional buttons appear on the page. Following are the buttons that appear and how you can use them to modify the command line:

- **UI Options**
  Make the installation run in silent mode and set the user interface level.
  See *Adding UI Options to Your Command Line* on page 100.

- **Logging Options**
  Generate a log when the installation is run and select logging options.
  See *Adding Logging Options to Your Command Line* on page 101.

- **Advertising Options**
  Advertise applications and apply a transform to the advertised package.
  See *Adding Advertising Options to Your Command Line* on page 102.

- **Repair Options**
  Repair installed files.
  See *Adding a Repair Option to Your Command Line* on page 103.

- **Edit Properties**
  Change the value of public properties.
  See *Editing Public Properties With a Command Line* on page 104.

- **Apply Transforms**
  Apply transforms to the installation package using the TRANSFORMS property.
  See *Applying Transforms With a Command Line* on page 104.

- **Add/Remove Patches**
  Add or remove patches using the PATCH or MSIPATCHREMOVE properties.
  See *Applying or Removing Patches With a Command Line* on page 104.

7. When you finish modifying the command line, click Next on the Define Command Line page.
   The Finish page appears, displaying the command line you built.

8. Do one of the following:
   - Click Execute to run the command line.
   - Click Create Shortcut to create a shortcut file (.LNK), specify the file, and click Save.
   - Click Copy Command Line to copy the command line to the clipboard.

9. Click Finish.
Adding UI Options to Your Command Line

The Command Line Builder lets you create a command line that sets UI options, which determine how much the end user interacts with the installation. See User Interface Levels in the Windows Installer SDK Help. You can set UI options for all versions of Windows Installer or for Windows Installer 3.0 only.

To add UI options to a command line

1. On the Define Command Line page, click UI Options.
   
   See Creating a Command Line With the Command Line Builder on page 98.
   
   The UI Options dialog box appears.

2. To enable the UI Options check boxes, mark Set User Interface level.

3. By default, qn - No UI is marked in the All Windows Installer versions section. To set user interface level options for all versions of Windows Installer, mark the appropriate options in this section.

   - qn - No UI
     Displays no user interface during the installation.

   - qb - Basic UI
     Displays built-in modeless dialog boxes that show progress messages during the installation.

   - qr - Reduced UI
     Displays authored modeless dialog boxes and built-in modal error-message boxes during the installation.

   - qf - Full UI
     Displays both modal and modeless dialog boxes that have been authored into the internal user interface, and built-in modal error-message boxes during the installation.

   - qn+ - No UI
     Displays no user interface, except for a modal dialog box at the end of the installation.

   - qb+ - Basic UI
     Displays built-in modeless dialog boxes that show progress messages during the installation and a modal dialog box at the end of the installation.

   - qb- - Basic UI
     Displays built-in modeless dialog boxes that show progress messages and no modal dialog boxes during the installation.

4. If you mark the qb, qb+, or qb- option, then the Hide the Cancel Button check box is enabled. Check this to add the ! switch to the command line, which removes the Cancel button from the installation dialog boxes.

5. To set User Interface level options for Windows Installer 3.0 only, mark one of the following options in the Windows Installer 3.0 only section. This overrides any options you mark in the All Windows Installer versions section.
- **Quiet - No UI**
  Displays no user interface during the installation.

- **Passive - display a progress bar but no other prompts or messages**

**Note**
These options are enabled only if Windows Installer 3.0 or later is installed on your computer. They will work on the destination computer only if it has Windows Installer 3.0 or later installed.

6. If you mark **Quiet** or **Passive** in the **Windows Installer 3.0 only** section, you can also select an option to control how Windows Installer handles restarts.

- **Default (Restart if required)**
  Restarts the computer when necessary with no prompts or warnings to the user.

- **No restart**
  Never restarts the computer after the installation.

- **Force restart**
  Always restarts the computer after the installation.

- **Prompt restart**
  (Passive only.) Displays a message that a restart is required to complete the installation and asks the user if they want to restart the computer now.

7. Click OK.
   The Define Command Line page reappears, and the selected silent option is added to the command line.

### Adding Logging Options to Your Command Line

The Command Line Builder lets you create a command line that sets logging options that determine what activities are logged during the installation. For information on logging options, see *Logging* in the Windows Installer SDK Help. You can set logging options for all versions of Windows Installer or for Windows Installer 3.0 only.

#### To add logging options to a command line

   See *Creating a Command Line With the Command Line Builder* on page 98.

   The Logging Options dialog box appears.

2. To enable the Logging Options check boxes, mark **Create Log File**.

   This enables the field to its right, which displays the default location for a log file created during installation. The default location is the Temp directory.

3. To change the location of the log file, specify a new path.

4. By default, **- Wildcard, log all information** is marked in the **All Windows Installer versions** section. When this option is selected, the first four options in this section are enabled. Clear this check box to enable all the other options in this section. To set logging options for all versions of Windows Installer, mark the appropriate options in this section.

   - **- Wildcard, log all information**
     Logs all information, but does not use verbose output.
v - Verbose output
Logs more detailed information about each event or error.

+ - Append to existing file
Appends the log to an existing log file.

! - Flush each line to the log

i - Status messages

w - Non-fatal warnings

a - Start up of actions
Logs actions as they are started.

r - Action-specific records

u - User requests

c - Initial UI parameters
Logs the initial user interface parameters.

m - Out-of-memory or fatal exit information

o - Out-of-disk-space messages

p - Terminal properties

e - All error messages

5. To set logging options for Windows Installer 3.0 or later, mark log - log all information in the Windows Installer 3.0 or later section. This overrides any options you mark in the All Windows Installer versions section.

Note
This option is enabled only if Windows Installer 3.0 or later is installed on your computer. It will work on the destination computer only if it has Windows Installer 3.0 or later installed.

6. Click OK.

The Define Command Line page reappears, and the selected logging options are added to the command line.

Adding Advertising Options to Your Command Line

Windows Installer can advertise the availability of an application to end users and to other applications without actually installing the application. If an application is advertised, only the interfaces required for installing the application are presented to the end user or other applications, saving time and disk space. End users install the application by activating the advertised interface.

The Command Line Builder lets you set advertising options that determine who sees the advertised application and whether a transform is applied to it. For information on advertising options, see Advertisement in the Windows Installer SDK Help.

To add advertising options to a command line

1. On the Define Command Line page, select Advertised from Install Mode.

See Creating a Command Line With the Command Line Builder on page 98.

The Advertising Options button appears.
2. Click Advertising Options.
   The Advertising Options dialog box appears.

3. Complete the dialog box:
   - m - Advertise to all users of machine
   - u - Advertise to the current user
   - t - Applies transform to advertised package
     Add a transform to the advertised installation.
     In the field below the check box, specify the transform file to include in the
     installation.

4. Click OK.
   The Define Command Line page reappears, and the selected advertising option and
   transform are added to the command line.

Adding a Repair Option to Your Command Line

The Command Line Builder lets you set a repair option for an installation that
determines the files that are reinstalled during a repair. You can also specify whether
files are rewritten, overwritten, or run from the source. For information regarding Repair
Options, see REINSTALLMODE Property in the Windows Installer SDK Help.

To add a repair option to a command line
1. On the Define Command Line page, select Repair from Install Mode.
   See Creating a Command Line With the Command Line Builder on page 98.
   The Repair Options button appears.

2. Click Repair Options.
   The Repair dialog box appears.

3. Complete the dialog box:
   - p - Reinstall only if file is missing
   - o - Reinstall if file is missing or if an older version is installed
   - e - Reinstall if file is missing or an equal or older version is installed
   - d - Reinstall if file is missing or a different version is installed
   - c - Reinstall if file is missing or the stored checksum doesn’t match the
     calculated value
   - a - Force all files to be reinstalled
   - u - Rewrite all required user specific registry entries
   - m - Rewrite all required machine specific registry entries
   - s - Overwrite all existing shortcuts
   - v - Run from source and re-cache the local package

4. Click OK.
   The Define Command Line page reappears, and the selected repair options are
   added to the command line.
**Editing Public Properties With a Command Line**

For information on public properties, see *Public Properties* in the Windows Installer SDK Help.

**To edit public properties with a command line**

1. On the Define Command Line page, select **Install** or **Network Install** from **Install Mode**.
   
   See *Creating a Command Line With the Command Line Builder* on page 98.
   
   The Edit Properties button appears.

2. Click Edit Properties.
   
   The Add Property Dialog dialog box appears.

3. Select a property in the left pane and click Add to copy it to the right pane.

4. In the right pane, enter the value for the property.

5. Click OK.
   
   The Define Command Line page reappears, and the edited properties are added to the command line.

**Applying Transforms With a Command Line**

For information on transforms, see *TRANSFORMS Property* in the Windows Installer SDK Help.

**To apply transforms with a command line**

1. On the Define Command Line page, select **Install** from **Install Mode**.
   
   See *Creating a Command Line With the Command Line Builder* on page 98.
   
   The Apply Transforms button appears.

2. Click Apply Transforms.
   
   The Transform List Builder dialog box appears.

3. Click Add and specify the transform file (.MST).
   
   The full path appears on Transform List Builder dialog box.

4. Repeat the preceding step to specify additional transforms.

5. Because Windows Installer applies transforms in the order specified, adjust the order of the transforms as needed.

6. Click OK.
   
   The Define Command Line page reappears, and the transforms are added to the command line.

**Applying or Removing Patches With a Command Line**

The Command Line Builder lets you create a command line that updates a package by applying or removing patches. For information on patches, see *PATCH Property* and *MSIPATCHREMOVE Property* in the Windows Installer SDK Help.
Prior to Windows Installer 3.0, you could only remove a patch by uninstalling the entire application. Beginning with Windows Installer 3.0, you can remove a single patch or a set of patches in any order without uninstalling the application. See Removing Patches and Uninstallable Patches in the Windows Installer SDK Help.

**To apply or remove patches with a command line**

1. On the Define Command Line page, select one of the following from **Install Mode**. See Creating a Command Line With the Command Line Builder on page 98.
   
   - **Install**
     This is available only if you specified a file in the **File Location** section on the Welcome page box. Use it to remove or apply patches. You can apply patches to an installed package or to the package being installed by the command line.
   
   - **Update**
     This is available only if you left the **File Location** section on the Welcome page blank. Use it to update installed applications.

   The Add/Remove Patches button appears.

2. Click Add/Remove Patches.

   The Patch List Builder dialog box appears.

3. In the **Add/Remove** section, specify whether to add or remove patches.

   **Note**
   The option to remove patches is enabled only if Windows Installer 3.0 or later is installed on your computer.

4. Click Add and specify a patch file (.MSP).

   The full path appears in the Patch List.

5. Repeat the preceding step to specify additional patches.

   **Note**
   Windows Installer 3.0 or later is required to add multiple patches.

6. Windows Installer applies patches in the order they are listed. To rearrange the order, click Move Up or Move Down. If you used patch sequencing with Windows Installer 3.0 when you created the patches, that sequencing would override the order you specify here.

7. Click OK.

   The Define Command Line page reappears, and the patches are added to the command line.

**InstallTailor**

Use InstallTailor™ to create a transform (.MST) that changes the way a Windows Installer installation runs. A transform is a special kind of Windows Installer database that can be applied at run time to a Windows Installer package to customize the installation for a particular group of end users.

When you run InstallTailor, it simulates an installation, captures the options that you select on the installation dialog boxes, and creates a transform file that incorporates
those selections. Because the installation is only simulated, no changes are actually made on your computer.

Examples:

- Set certain features to be installed for a particular group of end users.
- Change the default target directory for all users that install an application within the corporate environment.
- When an installation cannot run silently, turn off some of the dialog boxes so that end users do not provide inaccurate information during installation. The ability to turn off dialog boxes is not supported in some installations.
- Define information about the SQL Server that should be used by an application without prompting the end user.

The transform that InstallTailor creates can set the values of certain properties even if you do not explicitly define those values when you run InstallTailor. You can review the changes that InstallTailor captures, you can edit them, and you can delete the ones that should not appear in the transform.

See About InstallTailor Changes on page 106.

You can create a transform that customizes another transform. Example: Suppose you are deploying a new office automation application to your organization’s office in Germany. Everyone in that office needs the German-language version of the installation, but only the Sales department needs the application’s contact management component. First, you create a language transform to change the language of the installation dialog boxes to German. Then you use InstallTailor to create a second transform, based on the first transform, that installs the contact management component. When the second transform is run, it applies the first transform to the base installation, then applies the second transform.

Transfoms are not applicable to WiseScript technology. For general information on transforms, see About Transforms in the Windows Installer Editor Help.

See Creating a Transform with InstallTailor

About InstallTailor Changes

The transform that InstallTailor creates can set the values of certain properties even if you do not explicitly define those values when you run InstallTailor. For example, the base installation might set a property based on the destination computer’s operating system. However, the operating system on which you run InstallTailor might differ from the operating system on your target computers.

InstallTailor displays the changes that it captures when you simulate the installation and lets you do the following:

- Delete the changes that should not appear in the transform.
- Edit the captured changes.
- Add changes to property values, feature states, and directory values that InstallTailor did not capture.

For example, you want to specify a different installation directory, but the option to do so does not appear on the installation dialog boxes. You can add a value for the installation directory and it is added to the transform.
Creating a Transform with InstallTailor

Note
You cannot use InstallTailor™ to create a transform for an application that is already installed on your computer. Uninstall the application before you run InstallTailor.

To create a transform with InstallTailor

1. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with InstallTailor. This tool might skip pages or populate fields based on command-line options defined in Process Templates Setup.
   - On the Tools tab, double-click InstallTailor.

   The Welcome page appears.

2. Complete the page:
   - **Windows Installer package**
     Specify the installation file (.MSI) or transform file (.MST) to create a transform for.
   - **Base MST file**
     Specify a transform to apply to the installation.

3. Click Next.

   If you specified a transform file, you might be prompted to specify the base .MSI, if that information is not embedded in the transform.

   The installation runs in a simulated mode.

4. Step through the installation and make the changes to save in the transform. The resulting transform, when applied to the base installation, will make the same changes that you make during this simulated installation.

5. If the **Hide Dialog** check box is available on an installation dialog box, mark it to hide that dialog box from the end user during installation. This ensures that changes you make on the dialog box cannot be overridden by the end user.

   This check box does not appear on complex, custom dialog boxes, or for certain installations that do not support it.

6. When you complete the simulated installation, a message indicates that changes to the installation have been captured. Click OK.

7. On the Captured Changes page, review the changes that were made to the properties, features, and dialog boxes of the installation. Do any of the following and then click Next.
   - To exclude a change from the transform, uncheck its check box.
   - To hide all dialog boxes in the installation, check **Hide all dialogs**.
   - To edit a change, select it in the list and click Details.
     You cannot edit the changes that hide a dialog box.

See *Editing InstallTailor Changes* on page 108.

Any changes that you make to the captured changes appear in the transform that InstallTailor creates.
To add a value for a property, directory custom action, or feature, click Add and then select the appropriate command from the Add menu.

See Editing InstallTailor Changes on page 108.

8. On the Capture Complete page, enter the following:

If you are working with a process, Transform file name is set to [ProjectDir]\[ProjectName].mst, and Shortcut name is set to [ProjectName].

- Transform file name
  Specify the name and location for the new transform file (.MST).
  If you started InstallTailor from the Projects tab, this field might not be editable.

- Shortcut name
  Enter the name of the shortcut to run this transform file with the base installation. This defaults to the Transform file name specified above.
  If you started InstallTailor from the Projects tab, this field might not be editable.

- Edit this transform file using Windows Installer Editor
  Mark this to make changes to the transform that were not possible in InstallTailor. If you mark this, the transform file opens in Windows Installer Editor after you click Finish.
  If you started InstallTailor from the Projects tab, this check box might not appear.


The .MST and shortcut are created in the location you specified or in the current project’s directory. Double-clicking the shortcut applies the transform to the base installation.

See Applying a Transform to an Installation in the Windows Installer Editor Help.

Editing InstallTailor Changes

The Captured Changes page in InstallTailor displays the changes that InstallTailor captures when you simulate the installation.

See About InstallTailor Changes on page 106.

You can delete, edit, and add to the changes that InstallTailor captures.

To delete a captured change

- On the Captured Changes page, uncheck the check box next to the change to delete.

To hide all dialog boxes in the installation

- On the Captured Changes page, check Hide all dialogs.

To edit a change

- On the Captured Changes page, select the change in the list and click Details.
  You cannot edit the changes that hide a dialog box.
To add a property value to the transform

1. On the Captured Changes page, click Add and on the Add menu, click Property Value.

2. On the Property Value dialog box, enter the following and then click OK:

   - **Name**
     Select the property from the list, which contains all the properties that are in the installation but do not appear in the Captured Changes list.

   - **Default Value**
     (Read-only) The installation sets this value for this property unless you override it.

   - **Value**
     To override the default value or change the captured value, type a new value. To delete an existing value, leave this box empty.

To add a directory custom action to the transform


2. On the Directory Custom Action dialog box, enter the following and then click OK:

   - **Name**
     Select the directory from the list, which contains all the directories that are in the installation but do not appear in the Captured Changes list.

   - **Default Value**
     (Read-only) The installation sets this value for this directory unless you override it.

   - **Value**
     To override the default value or change the captured value, type a new value. To use a property, enclose the property name in brackets. For example, to set the directory path to a directory named Data in the Windows directory, enter the following:

     `[WindowsFolder]Data`

     To delete an existing value, leave this box empty.

To add a feature level to the transform

1. On the Captured Changes page, click Add and on the Add menu, click Feature State.

2. On the Feature State dialog box, enter the following and then click OK:

   - **Feature**
     Select the feature from the list, which contains all the features that are in the installation.

   - **Install State**
     Select the installation state to apply to this feature. The installation state determines the default state of the feature during installation.

See also:

*Creating a Transform with InstallTailor* on page 107
Legacy Setup Conversion

Use Legacy Setup Conversion to convert the following types of setup programs into Windows Installer packages:

- Microsoft SMS (.IPF or .EXE)
  See SMS Conversion Guidelines on page 110 and Converting an SMS Installation on page 111.

- Novell ZENWorks® (.AXT)
  See Novell Conversion Guidelines on page 112 and Converting a Novell Installation on page 113.

- WinINSTALL™ (.TXT)
  See WinINSTALL Conversion Guidelines on page 114 and Converting a WinINSTALL Installation on page 115.

- WiseScript (.WSE or .EXE)

- InstallShield® Professional (.IPR)
  See InstallShield Professional Conversion Guidelines on page 118 and Converting an InstallShield Professional Installation on page 118.

- InstallShield® (.MSI or .EXE)
  See InstallShield .MSI Conversion Guidelines on page 119 and Converting an InstallShield .MSI Installation on page 120.

- Altiris® RapidInstall Package (.EXE)
  See Altiris RapidInstall Package Conversion Guidelines on page 121 and Converting an Altiris RapidInstall Package on page 122.

Converting an installation file rather than capturing a setup program retains source paths and other pre-compile information that might not be available in a compiled setup program.

Depending on the format you are importing, some elements of the installation might not be converted. This typically is due to differences in technology between the source format and Windows Installer. To ensure that all elements of the installation are converted, use SetupCapture instead of Legacy Setup Conversion. However, source paths are not retained during SetupCapture.

See About Capturing Applications on page 201.

SMS Conversion Guidelines

Following are guidelines for using Legacy Setup Conversion to convert an installation (.IPF) that was created in the Microsoft System Management Server (SMS) to a Windows Installer package (.WSI or .MSI).

- You cannot convert all .EXEs—only those that were created with Microsoft SMS.
- If you do not have the original installation project file, you can convert the compiled .EXE instead, because it contains an embedded copy of the script.
- Because script installations are based on a different technology than Windows Installer, not all elements of the script are converted to Windows Installer format.
Only the installation of files, registry changes, and other system changes are converted.

Custom dialog boxes, custom logic, and other settings are not converted.

All the files that are available to the original installation must be available to the converted Windows Installer package at the same locations.

Components are converted to features, and Execute Program script actions are converted to Execute Program custom actions.

Some script actions can cause problems with the conversion process.

To edit an .IPF file without converting it to a Windows Installer package, open it in WiseScript Editor or WiseScript Package Editor. The WiseScript tools natively support Microsoft SMS files.

Converting an SMS Installation

To convert an SMS installation

1. Before you convert a script, open it and delete all Display Billboard, Display Graphic, and Add Icon script actions.

2. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with Legacy Setup Conversion. This tool might skip pages or populate fields based on command-line options defined in Process Templates Setup.
   - On the Tools tab, double-click Legacy Setup Conversion.

3. If the Select Source Format page appears, mark Microsoft® SMS Installer and click Next.

4. Complete the Specify Files page and click Next.

   - **Source Installation**
     Specify the full path of the .IPF or .EXE to convert.

   - **Target Installation**
     Specify the full path of a new or existing .WSI or .MSI in which to save the converted installation.

   - **Add/Update Resources in Existing Installation**
     If you specified an existing installation in **Target Installation**, you can mark this to add to or update the resources in the existing installation instead of overwriting them.

5. If you are converting an .EXE, the Specify Temporary Directory page appears. In **Extract Directory**, specify a directory in which to hold the extracted files and click Next.

     Files must be extracted from the .EXE before the conversion can begin. This becomes the source directory for the new package.

6. Click Next to start the conversion.

    The .IPF or .EXE is converted to Windows Installer format.

    When the conversion is finished, the Conversion Complete page appears. It shows the results of the conversion and lists any errors or problems that might have
occurred. (Example: An error occurs when files that were referenced by the source installation cannot be found.)

7. To obtain a record of any conversion errors, click Save Errors or Print Errors.

8. Click Finish.

More errors might appear at this point, which have to do with saving in Windows Installer format.

If a file that is part of a merge module is added, the Files in Merge Modules dialog box appears. It prompts you to add the merge module and, if necessary, download it.

See Adding Merge Modules Instead of Files on page 232.

If a file that is used by a package in the Wise Software Repository is added, the Files in Repository dialog box appears and prompts you to add a version of the file that is in the repository.

See Adding Files From the Wise Software Repository on page 144.

9. Open the resulting package in Windows Installer Editor to view the converted file and to resolve reported problems.

Pages in Installation Expert, such as the Files page, are populated based on the contents of the source installation. If you converted an .IPF, files are referenced from their original locations. If you converted an .EXE, files from the converted installation are stored in and referenced from the extract directory that you specified above.

**Novell Conversion Guidelines**

Following are guidelines for using Legacy Setup Conversion to convert an existing application object from Novell ZENworks 3.x or later to a Windows Installer package (.WSI or .MSI):

- Legacy Setup Conversion cannot convert a Novell application object directly; you must first convert the Novell application object to an .AXT (text-based) file. Legacy Setup Conversion then converts the text file to a Windows Installer package.

- All the files available to the original Novell installation must be available to the converted Windows Installer package at the exact same locations.

- Legacy Setup Conversion cannot convert the following sections of an .AXT file:
  
  [Application Icon]  
  [Application Icon Order]  
  [Application Description]  
  [Text File Add Begin]  
  [Filter *] (where * is not OS)

- The following sections of an .AXT file are not converted because they do not apply to a Windows Installer package:
  
  [Application Caption]  
  [Application Date]  
  [Application Time]  
  [Application Flags]  
  [Application Path]  
  [Application Parameters]  
  [Application Shutdown Script]
The Flag line in the [FileCopy] section of an .AXT file is ignored because it doesn’t apply to a Windows Installer package.

**Note**
If the Novell file you import was originally created with Novell snAppShot, the file extensions (.FIL) of source files may prevent file associations in COM registry keys from being assigned to the same component as the associated files. To correct this, change the file extension of the source files to .EXE, .DLL, or .OCX as appropriate. Then direct the files in the installation to the new source names either in Novell before the conversion or on the Files page in Windows Installer Editor after the conversion.

### Converting a Novell Installation

**To convert a Novell installation**

1. In Novell ConsoleOne, export the application object to an .AXT (text-based) file.
   
   Be sure to specify the extension .AXT; otherwise, the application defaults to an .AOT file, which cannot be converted.

   **Note**
   The source paths for installation files are hard-coded in the .AXT file. Therefore, if you move the .AXT file to a different computer, you must also move the directory containing the source files. Then, open the .AXT file in a text editor such as Notepad, find the section that contains Name=SOURCE_PATH, and change the Value of the source path to the new file location.

2. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with Legacy Setup Conversion. This tool might skip pages or populate fields based on command-line options defined in Process Templates Setup.
   - On the Tools tab, double-click Legacy Setup Conversion.

3. If the Select Source Format page appears, mark Novell® ZENworks® and click Next.

4. Complete the Specify Files page:
   - **Source Installation**
     Specify the full path of the .AXT file to convert.
   - **Target Installation**
     Specify the full path of a new or existing .WSI or .MSI. The converted installation will be stored in this file.
   - **Add/Update Resources in Existing Installation**
     If you specified an existing installation in Target Installation, you can mark this to add to or update the resources in the existing installation instead of overwriting them.

5. Click Next to start the conversion process.

   The .AXT file is converted to Windows Installer format.
When the process is finished, the Conversion Complete page appears. It shows the results of the conversion and lists any errors or problems that might have occurred. (Example: If files referenced by the source installation could not be found, an error is displayed.)

6. To obtain a record of the conversion errors, click Save Errors or Print Errors.

7. Click Finish.

More errors might appear at this point, which have to do with saving in Windows Installer format.

If a file that is part of a merge module is added, the Files in Merge Modules dialog box appears. It prompts you to add the merge module and, if necessary, download it.

See Adding Merge Modules Instead of Files on page 232.

If a file that is used by a package in the Wise Software Repository is added, the Files in Repository dialog box appears and prompts you to add a version of the file that is in the repository.

See Adding Files From the Wise Software Repository on page 144.

8. Open the resulting package in Windows Installer Editor to view the converted file and to resolve reported problems.

Pages in Installation Expert, such as the Files page, are populated based on the contents of the source installation.

### WinINSTALL Conversion Guidelines

Following are guidelines for using Legacy Setup Conversion to convert an existing OnDemand Software WinINSTALL™ installation to a Windows Installer package (.WSI or .MSI). In the Professional Edition, you also can convert a WinINSTALL package to a WiseScript (.WSE).

- Legacy Setup Conversion cannot convert a WinINSTALL installation (.NAI) or compiled executable directly; you must first convert your WinINSTALL installation to a text file (.TXT). Legacy Setup Conversion then converts the text file to a Windows Installer package or WiseScript package (Professional Edition only).
- All the files available to the original WinINSTALL installation project must be available to the converted installation at the exact same locations.
- Because WinINSTALL installations are based on a different technology than Windows Installer or WiseScript, not all elements of a WinINSTALL installation are converted:
  - Only the installation of files, registry changes, and other system changes are converted.
  - Custom logic written in the WinINSTALL custom scripting language is not converted.
  - WinINSTALL environment variable assignments are not converted. To re-add environment variable assignments in a Windows Installer installation, open it in Windows Installer Editor and use the Environment Variable icon, located on the Features and Components tabs in Setup Editor. To re-add environment variable assignments in a WiseScript, open it in WiseScript Package Editor (Professional Edition only) and use the Autoexec.bat page.
The WinINSTALL Preinstall and Postinstall scripts might not convert correctly. Legacy Setup Conversion tries to convert them to Execute Program custom actions in MSI Script in Windows Installer Editor or to Execute Program script actions in Script Editor in WiseScript Package Editor. However, the actions might not be configured properly. Check and test each of these Execute Program actions to see if it is configured correctly.

- If the target path of a file contains a WinINSTALL variable, then the WinINSTALL variable is converted to a Wise variable.
- If the source path of a file in WinINSTALL contains either the @Server or @Wininstall variable, you can specify the values of these two variables at conversion time.
- If the source path of a file contains other WinINSTALL variables, regardless of the target path, then the entire file is ignored by the conversion process. Any files with other WinINSTALL variables in the source path must be added manually to the converted installation. This problem does not occur if source paths in the WinINSTALL installation have hard-coded or UNC-formatted paths. However, the files must be available at the same locations as specified in the original WinINSTALL installation.

Converting a WinINSTALL Installation

To convert a WinINSTALL installation
1. In WinINSTALL, open the installation and export it to a text file. (In WinINSTALL 7 or WinINSTALL 2000, use the Export command on the Actions menu.)
   The text file has the same name as the WinINSTALL installation, but with the extension .TXT, and it is saved in the same directory.
2. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with Legacy Setup Conversion. This tool might skip pages or populate fields based on command-line options defined in Process Templates Setup.
   - On the Tools tab, double-click Legacy Setup Conversion.
3. If the Select Source Format page appears, mark WinINSTALL® and click Next.
4. Complete the Specify Files page:
   - **Source Installation**
     Specify the full path of the .TXT file to convert.
   - **Target Installation**
     Specify the full path of a new or existing .WSI or .MSI. In the Professional Edition, you also can specify a .WSE file. The converted installation will be stored in this file.
   - **Add/Update Resources in Existing Installation**
     If you specified an existing installation in Target Installation, you can mark this to add to or update the resources in the existing installation instead of overwriting them.
5. Click Next on the Specify Files page.
   The WinINSTALL Files page appears.
6. Complete the page:
- **Replace @Wininstall With**
  If source paths in the WinINSTALL text file contain the variable @Wininstall, specify the value of the variable here. The value you specify replaces all instances of @Wininstall located in source paths.

- **Replace @Server With**
  If source paths in the WinINSTALL text file contain the variable @Server, specify the value of the variable here. The value you specify replaces all instances of @Server located in source paths.

7. Click Next.
   The conversion process begins, and status messages and major errors appear in the Conversion Status section.

8. To obtain a record of the conversion errors, click Save Errors or Print Errors.

9. Click Finish to complete the conversion process.
   If a file that is part of a merge module is added, the Files in Merge Modules dialog box appears. It prompts you to add the merge module and, if necessary, download it.
   See *Adding Merge Modules Instead of Files* on page 232.

   If a file that is used by a package in the Wise Software Repository is added, the Files in Repository dialog box appears and prompts you to add a version of the file that is in the repository.
   See *Adding Files From the Wise Software Repository* on page 144.

10. Open the resulting package in Windows Installer Editor (.WSI or .MSI) or WiseScript Package Editor (.WSE) to view the converted file and to resolve reported problems.
    Pages in Installation Expert, such as the Files page, are populated based on the contents of the source installation.

### WiseScript Conversion Guidelines

Following are guidelines for using Legacy Setup Conversion to convert a WiseScript™ (.WSE) that was created in version 5.0 and later of Wise Installation System or in any version of WiseScript Editor or WiseScript Package Editor.

- If you do not have the original .WSE, you can convert the compiled .EXE instead, because it contains an embedded copy of the script. You cannot convert all .EXEs—only .EXEs that were created with one of the applications above.
- Because WiseScripts are based on a different technology than Windows Installer, not all elements of your script are converted to Windows Installer format:
  - Only the installation of files, registry changes, and other system changes are converted.
  - Custom dialog boxes, custom logic, and other settings are not converted.
- All the files available to the original WiseScript installation must be available to the converted Windows Installer installation at the same locations.
- Components are converted to features, and Execute Program script actions are converted to Execute Program custom actions.
- Some script actions can cause problems with the conversion process.
Converting a WiseScript

To convert a WiseScript

1. Before converting a script, open it and delete all Display Billboard, Display Graphic, and Add Icon script actions from the script.

2. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with Legacy Setup Conversion. This tool might skip pages or populate fields based on command-line options defined in Process Templates Setup.
   - On the Tools tab, double-click Legacy Setup Conversion.

3. If the Select Source Format page appears, mark WiseScript and click Next.

4. Complete the Specify Files page:
   - **Source Installation**
     Specify the full path of the .WSE or .EXE to convert.
   - **Target Installation**
     Specify the full path of a new or existing .WSI or .MSI in which to save the converted installation.
   - **Add/Update Resources in Existing Installation**
     If you specified an existing installation in Target Installation, you can mark this to add to or update the resources in the existing installation instead of overwriting them.

5. Click Next on the Specify Files page.

6. If you are converting an .EXE, the Specify Temporary Directory page appears. In Extract Directory, specify a directory to hold the extracted files and click Next.

   Files must be extracted from the .EXE before the conversion can begin. This becomes the source directory for the new installation.

7. The .WSE or .EXE is converted to Windows Installer format.

   When the conversion is finished, the Conversion Complete page appears. It shows the results of the conversion and lists any errors or problems that might have occurred. (Example: If files referenced by the source installation could not be found, an error is displayed.)

8. To obtain a record of the conversion errors, click Save Errors or Print Errors.

9. Click Finish on the Conversion Complete page.

   More errors might appear at this point, which have to do with saving in Windows Installer format.

   If a file that is part of a merge module is added, the Files in Merge Modules dialog box appears. It prompts you to add the merge module and, if necessary, download it.

   See Adding Merge Modules Instead of Files on page 232.

   If a file that is used by a package in the Wise Software Repository is added, the Files in Repository dialog box appears and prompts you to add a version of the file that is in the repository.

   See Adding Files From the Wise Software Repository on page 144.
10. Open the resulting package in Windows Installer Editor to view the converted file and to resolve reported problems.

Pages in Installation Expert, such as the Files page, are populated based on the contents of the source installation. If you converted a .WSE, files are referenced from their original locations. If you converted an .EXE, files from the converted installation are now stored in and referenced from the directory you specified in Extract Directory.

InstallShield Professional Conversion Guidelines

You can use Legacy Setup Conversion to convert an installation script project (.IPR) that was created in version 5.5 or later of InstallShield® Professional to a Windows Installer package.

Guidelines

- If you do not have access to the script project file but you do have an installation .EXE created in InstallShield Developer version 7 or 8, you can import it directly into the Software Manager database without repackaging it. For more information, see Importing an InstallShield Developer Executable in the Software Manager Help.

- Because InstallShield Professional installations are based on a different technology than Windows Installer, not all elements of the source installation are converted to Windows Installer format:
  - The installation is built from the configuration files defined by the .IPR. Files, registry keys, and shortcuts are converted. The Product Name, Product Version, and Company Name Properties are also converted. File source paths are retained.
  - The .RUL file, which controls how the resources are installed, cannot be converted. Add custom actions to the new package to replace any custom logic in the .RUL file.
  - Only InstallShield projects that use English in their dialog boxes can be converted.
  - All the files available to the original InstallShield installation must be available to the converted Windows Installer package at the same locations.
  - Resources are organized into features based on groups in the resource lists. To retain the organization in the InstallShield project, components are organized into separate features, with FileGroups as child features.

Converting an InstallShield Professional Installation

To convert an InstallShield Professional installation

1. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with Legacy Setup Conversion. This tool might skip pages or populate fields based on command-line options defined in Process Templates Setup.
   - On the Tools tab, double-click Legacy Setup Conversion.
2. If the Select Source Format page appears, mark **InstallShield® Professional** and click Next.

3. Complete the Specify Files page:
   - **Source Installation**
     Specify the full path of the .IPR file to convert.
   - **Target Installation**
     Specify the full path of a new or existing .WSI or .MSI in which to save the converted installation.
   - **Add/Update Resources in Existing Installation**
     If you specified an existing installation in **Target Installation**, you can mark this to add to or update the resources in the existing installation instead of overwriting them.

4. Click Next to start the conversion.
   The Conversion Progress page appears. The .IPR file is converted to Windows Installer format.

   When the conversion is finished, the Conversion Complete page appears. It displays the results of the conversion and lists any errors or problems that might have occurred. (Example: If files referenced by the source installation could not be found, an error is displayed.)

5. To obtain a record of the conversion errors, click Save Errors or Print Errors.

6. Click Finish.
   More errors might appear at this point, which have to do with saving in Windows Installer format.

7. Open the resulting package in Windows Installer Editor to view the converted file and to resolve reported problems.

   Pages in Installation Expert, such as the Files page, are populated based on the contents of the source installation.

   If a file that is part of a merge module is added, the Files in Merge Modules dialog box appears. It prompts you to add the merge module and, if necessary, download it.

   See **Adding Merge Modules Instead of Files** on page 232.

   If a file that is used by a package in the Wise Software Repository is added, the Files in Repository dialog box appears and prompts you to add a version of the file that is in the repository.

   See **Adding Files From the Wise Software Repository** on page 144.

### InstallShield .MSI Conversion Guidelines

You can use Legacy Setup Conversion to convert an .MSI that was created with InstallShield® Developer 7.0 or later to a non-proprietary .WSI or .MSI. This includes .MSIs that are wrapped in an InstallShield setup.exe.

This conversion:
- Strips the InstallShield branding from the installation dialog boxes
- Exposes self-registration information
• Exposes SQL Server configuration information

Guidelines
• If you try to convert an .EXE and the conversion fails, use SetupCapture to repackage the installation. The conversion could fail because the .EXE:
  • Is not an InstallShield .EXE.
  • Does not contain an .MSI.
• If you try to convert an .MSI and the conversion fails, use Windows Installer Editor to customize the package.
• After conversion, check the log file to see if any parts of the conversion failed and thoroughly test the converted file.
• If an InstallShield .MSI configures a SQL Server database during installation, the conversion of the .MSI has the following limitations:
  • If the InstallShield .MSI is installed on the computer where it is being converted, then its SQL Server scripts do not convert properly. To resolve this problem, either open the converted .MSI in Windows Installer Editor and replace the corrupt SQL scripts on the SQL Server Scripts page, or redo the conversion of the .MSI on a computer where the .MSI is not installed.
  • If the InstallShield .MSI specifies text strings to be found and replaced within SQL statements during installation, these are not converted. You must enter these in Windows Installer Editor on the SQL Server Scripts page.
  • If the InstallShield .MSI recreates a SQL Server database during installation and does not use a SQL script to do this, this is not converted. You must specify how to recreate the database in Windows Installer Editor on the SQL Server Scripts page.
• You can apply InstallShield patches and updates to a converted InstallShield .MSI.

Converting an InstallShield .MSI Installation

To convert an InstallShield .MSI installation
1. On the Tools tab, double-click Legacy Setup Conversion.
   The Select Source Format page appears.
2. Mark InstallShield (.MSI or .EXE) and click Next.
   The Specify Files page appears.
3. Complete the page:
   • Source Installation
     Specify the full path of the .MSI or the .EXE to convert. An .EXE must contain an .MSI.
   • Target Installation
     Specify the full path of a new or existing .WIS or .MSI in which to save the converted installation. Specify an empty directory because multiple files and directories may be generated.
Open the converted installation in Windows Installer Editor.
Mark this option to open the converted installation in Windows Installer Editor when the conversion is finished. You must use the Target Installation Browse button to enable this option.

4. To remove certain parts of the installation from the conversion process, click Advanced on the Specify Files page.

The Items to Convert page appears.

a. Clear the items you do not want to convert. This does not remove these items from the converted installation.

   ♦ Basic User Interface
   If you mark this option, the graphics on the installation dialog boxes and references to InstallShield in the text of the dialog boxes are converted.

   ♦ Customized User Interface
   If you mark this option, any default InstallShield installation dialog boxes that were renamed or any dialog boxes that were added to the default dialog boxes are converted.

   ♦ Self-Registration
   If you mark this option, self-registration information is converted and appears on the Self-registration tab of the File Details dialog box on the Installation Expert > Files page in Windows Installer Editor.

   ♦ SQL Server Operations
   If you mark this option, SQL Server scripts that configure a SQL Server during installation are converted to the Windows Installer Editor format. This information appears on the Installation Expert > SQL Server Scripts page in Windows Installer Editor.

b. Click Apply.

5. Click Next to start the conversion.

Several progress dialog boxes appear. The .MSI or .EXE is converted to a non-proprietary Windows Installer format (.MSI or .WSI).

When the conversion is finished, the Conversion Results page appears. It displays the results of the conversion and lists any problems that might have occurred. (Example: If the .EXE did not contain an .MSI, the conversion fails.)

6. To see a log of the conversion process, click View Log File.

Use the log file to verify that all parts of the conversion process were successful.

7. Click Finish.

If the conversion is successful and you marked the option on the Specify Files page to open the converted installation in Windows Installer Editor, it now opens.

Test the converted .MSI or .WSI to verify that all parts of the installation were converted successfully. If it needs editing, edit it in Windows Installer Editor.

**Altiris RapidInstall Package Conversion Guidelines**

You can use Legacy Setup Conversion to convert an Altiris RapidInstall package (.EXE) version 3.0 or later to an .MSI.
**Guidelines**

- If default values were not provided for each user defined variable in a RapidInstall package (RIP), the converted .MSI package may fail.
- If a RIP did not specify a package title, a title is generated for the .MSI package using the file name of the RIP.
- If a RIP specifies “Always replace” or “Never replace” file replacement behavior for non-versioned files, these specifications are not converted in the .MSI.
- If a RIP has an edit password to prevent unauthorized editing of the package, you must supply this password when converting the package.
- The RapidInstall “IP Address” built-in variable is not supported with .MSIs.

**Converting an Altiris RapidInstall Package**

**To convert an Altiris RapidInstall package**

1. On the Tools tab, double-click Legacy Setup Conversion.  
   The Select Source Format page appears.
2. Mark *Altiris RapidInstall Package (.EXE)* and click Next.  
   The Welcome page of the Rip to MSI Migration Utility appears.
3. Click Next.  
   The RIP Files to Migrate page appears.
4. To select the RapidInstall packages (RIPs) to convert, do one of the following:
   - Drag and drop RIPs from an explorer window.
   - Click Add.
5. If you clicked Add, complete the File Properties page and click OK:
   - **Source RIP Path**  
     Specify the RIP file to convert.
   - **Destination MSI Path**  
     Specify the name and location for the .MSI file. The converted installation is stored in this file.
   - **RIP Edit Password**  
     If the RIP file has an edit password to prevent unauthorized editing, enter it here.
   - **Migrate Add/Remove Program Data**  
     Mark this option to include Add/Remove Program data that was added by the RIP. This will cause duplicate entries in the Add/Remove Programs.
6. Repeat the previous steps to convert additional RIPs.
7. To edit an entry on the RIP Files to Migrate page, double-click it.
8. Click Next to begin conversion.
   When the files are converted, the Migration Summary page appears where you can review the Info, Warning, and Error messages for the conversion process.
9. Click Finish.
**Package Definition**

> Not available in Standard Edition.

A Wise package definition file defines what is needed to install a package. At a minimum, this is a command line. However, it can also be the installation file itself, additional command lines, or any file that needs to be installed. Having a package definition file lets you use Group Distribution in Software Manager to prepare the package for deployment.

**Defining a package in Workbench**

Create a package definition so you can install:

- A package that you could not otherwise repackage or deploy. Examples:
  - A Setup.exe of unknown format and the command line to run it.
  - An installation that you cannot repackage because its license agreement prohibits it.
  - A device driver (.INF) with its .CAB files and the command lines to run it.
  - An installation that is on one or more CDs (example: Microsoft Visual Studio .NET). In the package definition, specify the installation file or files and the command lines to run them.
- A package that consists of one or more command lines without any files. Example: Define a package that contains a command line that runs a shell command on the destination computer.

Package Definition creates a package definition file (.WPF), which lists the files and command lines that make up the package. The package definition file is saved in its own subdirectory of the share point Projects directory. The files in the package definition are saved in a Files subdirectory of the package definition’s project directory. This ensures that all the files are in a shared location and are accessible when the package is distributed.

Package Definition imports the package definition file or queues it for import into the Software Manager database. When the package definition file is imported into the Software Manager database, its resources are not imported and are not available for conflict management. To associate the resources with the package after the package definition is imported:

1. Use SetupCapture to capture the package installation as a Windows Installer file.
2. Import the Windows Installer file into the Software Manager database. During import, select the application and package name of the package definition file and clear the option to overwrite the package.

See *Creating a Package Definition File* on page 124.

**Defining a package in Software Manager**

In Software Manager, you can edit an existing package definition or create a new one. Create a package definition in Software Manager when you need to:

- Install a package that you could not otherwise repackage or deploy.
- Add a file or files to a package that is already in the Software Manager database.
- Change a command line for a package that is already in the Software Manager database.

See About Package Definition in the Software Manager Help.

See also:

Process for Deploying a Group in the Software Manager Help.

Creating a Package Definition File

Not available in Standard Edition.

A package definition file must contain at least one command line. Files are optional.

Note

Files that you add when you define the package do not appear in Software Manager. Therefore, you cannot resolve conflicts between files you add to the package definition and files in other packages.

To create a package definition

1. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with Package Definition. Package Definition will be run on the default project file. This tool might skip pages or populate fields based on command-line options defined in Process Templates Setup.
   - On the Tools tab, double-click Package Definition.
   The Specify Package page appears.

2. Complete the page:
   - **Project Name**
     Enter a name to identify the project when the package definition is saved in the share point Project directory.
   - **Application Name**
     Enter a name to identify this application in the Software Manager database.
   - **Package Name**
     Enter a unique name to identify this package in the Software Manager database. Typically, you use the application name plus specific version information. Example: If the application name is Adobe Acrobat Reader, the package name might be Acrobat Reader 9.0.
   - **Package Description**
     Describe the package. This becomes part of the package meta data when the package is imported into the Software Manager database.

3. Click Next on the Specify Package page.
   The Specify Files page appears. The upper panes of this page display the directories and files that are accessible from your computer.

4. (Optional.) To include files in the package definition, use the following buttons.
- **Add Directory**
  Adds the entire contents of the directory you select in the upper-left pane. If you select a file in the upper-right pane, it adds the entire contents of that file's directory. Files and directories that are in the exclusions list are not added. The files with their paths appear in the lower pane. Because Package Definition uses relative paths, it removes all drive letters from the directories. When the package is distributed, the files and directories are relative to the package's main directory.

- **Add File**
  Adds the files you select in the upper-right pane. The files appear in the lower pane with no path. When the package is distributed, the files are relative to the package's main directory.

  If a file is in the exclusions list, you are prompted to add the file anyway. Adding the file does not change the exclusions list. If you mark *Do not ask me again*, this page does not reappear for any of the other files that you selected that are in the exclusions list.

- **Exclusions**
  Opens the File and Folder Exclusions page, where you can specify the files to be excluded from this package. You exclude files to reduce the size of the package for distribution. Example: Exclude readme files or a help directory.

  See *Setting Exclusions in Package Definition* on page 127.

- **Delete**
  Deletes files that are selected in the lower pane.

  **Note**
  If you add an .MSI, .MSP, or .MST that requires other files, those files are added also. Example: If you add an .MSI that has external .CAB files, the .CAB files are added also.

5. Click Next on the Specify Files page.
   The Specify Command page appears.

6. To add a command line, click Add.
   The Specify Command Line page appears.

7. This step depends on the file that you are creating the command line for:

<table>
<thead>
<tr>
<th>If the file is this type</th>
<th>Then do this</th>
</tr>
</thead>
</table>
| An .MSI that is in the package definition | • Click Select File.  
  • Select the .MSI and click OK. The Command Line Builder for Windows Installer opens.  
    See *Command Line Builder* on page 97.  
  • Use the Command Line Builder to build the command line and click OK. When the command line appears in the Command Line Builder, it includes the drive letter for the file. Because Package Definition uses relative paths, it removes the drive letter. |
8. Click OK on the Specify Command Line page.
   The Specify Command page reappears and lists the command line you created.

9. Continue to add or delete command lines as needed.

10. The command lines are applied in the order they are listed. To rearrange the order, click Move Up or Move Down.

11. Click Next on the Specify Command page.
   The Finish page appears. The Package Definition Summary describes where the project and its files will be saved.

12. On the Finish page:
   - **Import into Software Manager**
     Mark this to import the package definition file into the Software Manager database. Clear this to have the package definition file queued for import into the database.
   - **Datasource**
     If you have multiple Software Manager databases, you can select the database to import to.

13. Click Finish.
   The package definition file is saved in its own subdirectory in the share point Projects directory. All the files that you specified are copied to a Files subdirectory of the package definition directory.

---

<table>
<thead>
<tr>
<th>If the file is this type</th>
<th>Then do this</th>
</tr>
</thead>
</table>
| A Microsoft hotfix that is in the package definition (For information on Microsoft hotfixes, search for "Command-Line switches for Windows software update packages" at msdn.microsoft.com.) | • We recommend that you first run the hotfix with the /? command-line option to display a list of the command-line options for that particular hotfix. Otherwise, you might choose command lines that are not appropriate for your hotfix.  
• Click Select File.  
• Select the Microsoft hotfix file and click OK.  
• On the Hotfix Patch Command Lines page, select command-line options and click OK. |
| Not an .MSI or hotfix OR Not in the package definition | • Enter the command line on the Specify Command Line page.  
• To add the name of a file that is in the definition, click Select File and select it.  
Because Package Definition uses relative paths, use relative paths in the command line. If you specify a directory, the command line will look for the file in that directory on the destination computer. |
Setting Exclusions in Package Definition

Not available in Standard Edition.

You can specify files and directories to be ignored by Package Definition, which reduces the size of the package for distribution. You can exclude:

- A file.
- A directory. You can exclude the files at the top level of the directory only, or all contents of the directory and its subdirectories.
- Files in a directory based on a wildcard. (Example: *.tmp for all temporary files.)

The default Package Definition exclusion list is the same as the default SetupCapture exclusion list, except that it only includes files.

To set file exclusions

1. Access the Specify Files page in Package Definition.
   See Creating a Package Definition File on page 124.
2. Click Exclusions.
   The File and Folder Exclusions page appears.
3. Click Add.
   The File Exclude page appears.
4. To exclude a file:
   a. In **File/Wildcard**, specify a file.
   b. **Directory** is pre-filled when you specify a file. You can use environment variables surrounded by percent signs (%) to specify paths. To exclude the file for all local drives, leave this field blank.
5. To exclude a directory:
   a. In **Directory**, specify a directory. This causes Package Definition to ignore files in the top level of this directory. You can use environment variables surrounded by percent signs (%) to specify paths.
   b. Leave **File/Wildcard** blank.
6. To exclude a file based on a wildcard:
   a. In **Directory**, specify a directory. The wildcard you set will apply to files in this directory. You can use environment variables surrounded by percent signs (%) to specify paths. To exclude the wildcard for all local drives, leave this field blank.
   b. In **File/Wildcard**, enter a wildcard.
7. To ignore files that match these criteria in all subdirectories of the directory you specified, mark **Exclude Sub-Directories**.
8. Click OK.
   The File and Folder Exclusions page reappears and contains the exclusions you added.
Note
If you exclude a file or directory that is under a user profile, the user profile name is replaced with a variable that always represents the current user profile name.

9. To apply the changes to this package definition file only, mark the check box at the bottom of the page.
   If you clear this check box, any changes you make on this page are saved in the exclusions list and will affect exclusions for future package definitions.

10. To edit an exclusion, double-click it in the list.

11. Click OK on the File and Folder Exclusions page.

Patch Creation

Use Patch Creation to create a Windows Installer patch file (.MSP) that updates installed versions of a Windows Installer-based application. A patch file can update one or several previous versions. Unlike full installations, a patch installation contains only the information necessary to update an installed version of the application.

Patch Creation uses three files from the Microsoft Windows Installer SDK: PATCHWIZ.DLL, MSPATCHC.DLL, and MAKECAB.EXE. They are installed in the Windows Installer Editor application directory. You specify settings in Patch Creation, and it saves those settings in a patch settings file (.PCP). Patch Creation then sends the patch settings file to PATCHWIZ.DLL, which creates the patch file (.MSP).

Note
Patch Creation is only applicable to Windows Installer (.MSI) technology, not WiseScript technology. For patching WiseScript installations, use the SmartPatch page in Installation Expert in WiseScript Package Editor.

Patch Creation guidelines

- Before you use Patch Creation, use UpgradeSync. UpgradeSync compares your current package with the previous version of the package, and prepares the current package for a patch or upgrade. See UpgradeSync on page 138.
- The package code must be different between the old installations and the updated installation. To see the package code in Windows Installer Editor, click the Summary icon in Setup Editor > Product tab. Changing the Version field on the Product Details page causes the package code to be updated, as does running UpgradeSync.
- The previous version or versions must have been installed using Windows Installer.
- To edit an existing patch, you need access to its patch settings file (.PCP).
- If other patches exist for this package, you need access to the most recent patch file (.MSP) to read its file sequence number and disk ID.
- You need access to the complete installation in .MSI format for every installation that this patch will upgrade and for the latest version of your application. If the installation package includes external compressed or uncompressed files, you need them also.
If you compiled the installation as an .EXE, you need the .MSI for the installation because Patch Creation does not operate on .EXE files. The .MSI is created in the same directory as the .EXE during compile.

**Patching assemblies in the Global Assembly Cache**

*Windows Installer 3.0 or later.*

Prior to Windows Installer 3.0, it was difficult to patch assemblies that were installed in the Global Assembly Cache (GAC), because the files could not be found.

When you patch an assembly that is installed in the GAC, Windows Installer 3.0 identifies and finds the original file and applies a binary patch. This eliminates the need for Windows Installer to access the original installation source in order to patch an assembly installed in the GAC. See `MsiPatchOldAssemblyName Table` and `MsiPatchOldAssemblyFile Table` in the Windows Installer SDK Help.

For additional topics, see *About Upgrading Applications* in the Windows Installer Editor Help.

**About Patch Sequencing**

*Windows Installer 3.0 or later.*

Patch sequencing ensures that patches are applied in the correct order, regardless of the order in which they are actually provided to the destination computer.

Sequencing is supported for small update patches only. Sequenced patches can be installed by Windows Installer 2.0, but the sequencing is ignored.

You add sequencing information to a patch during Patch Creation. You can add sequencing to a patch you created previously. Step through Patch Creation, open an existing patch file, enter sequencing information, and complete the wizard to recompile the patch.

**Order in which patches are applied**

1. Patches without sequence information are applied in the order they are provided to the destination computer. Example: If Patch2 is provided before Patch1, they will be applied in that order.

2. Sequenced patches are applied in sequence order.

**Patch families**

A patch family is a group of patches that update the same, similar, or related functionality of the application and should be applied in a specific order relative to other patches in the same family. Most patches will belong to a single family, and most applications will be updated by a single family. However, a patch can belong to multiple families if it applies to more than one application or includes multiple fixes.

Example: Suppose you have two applications that are sold separately, but are also sold together as a suite. Patches A and C only update Application1 and belong to family 100. Patch B only updates Application2 and belongs to family 200. Patch D updates both applications and belongs to family 100 and family 200.

You might sequence these patches as follows:
Creating a Patch File

Before you create a patch file, review the guidelines for patch creation.

See Patch Creation on page 128.

To create a patch file

1. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with Patch Creation. This tool might skip pages or populate fields based on command-line options defined in Process Templates Setup.
   - On the Tools tab, double-click Patch Creation.
     The Welcome page appears, listing the basic steps for creating a patch file. The wizard guides you through each step.

2. Click Next.
   The Specify Patch Settings File page appears.

3. Mark one of the following:
   - Create a new patch file
     This creates a new patch settings file (.PCP).
   - Open an existing patch settings file (.PCP file)
     If you mark this, also specify the .PCP file.

4. Click Next.
   The Specify Previous Versions page appears, where you select .MSI files of previous versions that this patch will update, referred to as targets. When this patch is run on a destination computer, it verifies that a valid target exists before installation. You must add at least one previous version to this list.

<table>
<thead>
<tr>
<th>Patch</th>
<th>Updates</th>
<th>Family</th>
<th>Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Application1</td>
<td>100</td>
<td>10</td>
</tr>
<tr>
<td>B</td>
<td>Application2</td>
<td>200</td>
<td>10</td>
</tr>
<tr>
<td>C</td>
<td>Application1</td>
<td>100</td>
<td>20</td>
</tr>
<tr>
<td>D</td>
<td>Application1</td>
<td>100</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>Application2</td>
<td>200</td>
<td>20</td>
</tr>
</tbody>
</table>

These patches would be applied to the applications in this order:

- Family 100: A, C, D
- Family 200: B, D

See also:

Specifying the Patch Sequence on page 136
Patch Creation on page 128
Sequencing Patches and MsiPatchSequence Table in the Windows Installer SDK Help
5. To add a previous version, click Add, complete the Previous Version Details page, and click OK.

See Specifying Previous Versions for Patches on page 133.

6. If you are prompted to run an administrative installation, click Yes.

The patch creation process, which is executed by the Microsoft-provided file PATCHWIZ.DLL, operates on uncompressed files only. Typically, the files in an .MSI are compressed. The administrative installation places all the files of the installation in a temporary directory. After the administrative installation finishes, the Specify Previous Versions page appears again.

**Note**
The Windows Installer engine is called to perform the administrative installation. If the operation fails for any reason, the error message generated by Windows Installer is displayed. To work around the error, open the .MSI in Windows Installer Editor, set it to generate uncompressed files, and recompile. Then specify the uncompressed version on the Specify Previous Version page. To set an .MSI to generate uncompressed files, go to the Media page, click Details, and in Compression Option select Uncompressed external files.

7. Repeat the steps above to add additional previous versions.

8. When you finish, click Next on the Specify Previous Versions page.

The Specify Upgrade Version page appears.

9. Complete the page:

   - **Upgrade MSI path**
     The earlier versions of your application will be upgraded to the version you specify here. By default, the path to the current installation's .MSI appears.

   - **Advanced**
     Click this to enter advanced settings. The Advanced Upgrade Version Details dialog box appears. Complete the dialog box and click OK.

     See Advanced Upgrade Version Details on page 134.

   - **Add a Digital Signature to the patch**
     (Windows Installer 3.0 or later.) Mark this to digitally sign the patch.

10. Click Next on the Specify Upgrade Version page.

11. If you are prompted to run an administrative installation again, click Yes. If you are prompted to update the package code, click Yes.

    If you marked the option to add a digital signature, the Specify Digital Signature Settings page appears. A warning message appears if the original installation was not signed.

12. To add a digital signature to the patch, complete the Specify Digital Signature Settings page.

    See Adding a Digital Signature to a Patch on page 135.

    If Windows Installer 3.0 or later is installed on your computer, the Patch Sequencing page appears. Complete the page and click OK. Otherwise, the Compile Patch page appears.

    See Specifying the Patch Sequence on page 136.
13. Complete the Compile Patch page:

- **Output .MSP file**
  Specify a full path for the patch file that you distribute to end users.

- **Advanced Settings**
  Click Advanced to display the Advanced Patch Settings dialog box. Complete the dialog box and click OK.

  See **Specifying Advanced Patch Settings** on page 137.

- **Patch Removal**
  (Windows Installer 3.0 or later only.) To make this patch removable through Add/Remove Programs, click Allow Removal and complete the Patch Removal Settings dialog box.

  See **Specifying Patch Removal Settings** on page 138.

- **Multi-patch Media Settings**
  During patch creation, entries are made in the Media table of the patch installation. The following options populate the Media table. For each subsequent patch, the file sequence start number and the disk ID start number must be higher than the one in the previous .MSI or patch file. To enter these numbers accurately, you must have access to the most recent patch file distributed to end users.

  - **File Sequence Start, Disk ID Start**
    The file sequence number indicates which files in the File table are located on a particular source disk. The disk ID number indicates how many media entries are in the Media table. For patches, these numbers must be at least 1 greater than the corresponding numbers in the most recent patch or .MSI for this application installation. To specify these numbers, browse to the most recent patch file created for this installation.

  - **Volume Label**
    Enter the name of the CD or other media on which this patch will ship. If the application needs repair in the future, Windows Installer uses this label to tell the end user what media to insert to perform the repair. If the patch is not shipped on any media but is distributed over the Internet, this field is ignored.

  - **Disk Prompt**
    Enter the prompt that the end user should see if this application needs to be repaired. Example: Insert the disk labeled Application 2.0.

14. Click Next to begin the patch creation process.

  During patch creation, you might see a message stating that the versions between the target image (previous version) and upgrade image (new version) do not match. This is normal; click Yes if this message appears.

  **Note**
  If the date and time of a file in the upgrade is earlier than the date and time of the matching file in the original installation, Patch Creation changes the date of the file in the upgrade to be later than that of the original installation.

  The Compile Patch page notifies you when patch creation is completed.

15. Click View Log to view a log file of all actions performed by PATCHWIZ.DLL to create the patch.
If the patch file could not be created, use this log file to determine the source of the error.

See also:
- *Patch Creation* on page 128
- *About Patch Sequencing* on page 129
- *Removing Patches* in the Windows Installer SDK Help

## Specifying Previous Versions for Patches

**To specify previous versions for patches**

1. Access the Previous Version Details page.
   
   See *Creating a Patch File* on page 130.

   This page is not related to the System Search page in Installation Expert, which lets you search for files, registry entries, and components, or search .INI files.

2. Complete the page and click OK:
   - **Previous MSI path**
     Specify the path to an .MSI that can be upgraded by this patch. You must specify an .MSI, even if you deployed the installation as a single-file .EXE.
   
   - **Ignore missing files while making patch**
     Mark this if you have files that you do not want to put in the patch, such as a ReadMe file. This lets you delete files from the .MSI without receiving an error during patch file creation.
   
   - **Match Product Code**
     Mark this if this patch should be installed only if the product code of the installed application on the destination computer matches the product code of the current installation. The product code is located on the Product Details page in Windows Installer Editor.
   
   - **Match Upgrade Code**
     Leave this marked to make sure that the upgrade code is the same for the previous application and the upgrade application. The upgrade code should always be the same for different versions of the same product.
   
   - **Match Language**
     Mark this to confirm that the language codes match between the target and upgrade images.
   
   - **Version To Check**
     Select what parts of the product version—the major, minor, or update version—should be used in the version relationship comparison. Example: Suppose the product version is 2.4.6801. The first segment is the major version; the second segment is the minor version; and the third segment is the update version (also called build version).

     The version for each installation is in the **Version** field on the Product Details page in Windows Installer Editor.
   
   - **Version Relationship**
     Select a comparison that must be true in order for the patch to be installed. Base Version is the version of the package you use to create the patch; this is the version you specify in **Previous MSI path**. Installed Version is the version
of the package being upgraded. In most cases, you should select the relationship **Base Version must be = Installed Version**. This means that the previous version you used to create this patch must match the version installed on the destination computer.

- **C++ Symbol File Directories (Optional)**
  If your application is written in C++, you can make your patch files smaller by specifying the directories where your Symbol and Object files are stored for the previous version of your application.

**Advanced Upgrade Version Details**

**To specify advanced upgrade version details**

1. Access the Advanced Upgrade Version Details dialog box.
   
   See *Creating a Patch File* on page 130.

2. Complete the dialog box and click OK:
   
   - **Patch GUID**
     Each patch file is assigned a GUID, independent of product codes and upgrade codes. The installation uses this to track which patches have been applied to an application. To change this value, click Generate or paste another valid GUID into this field.

     See *About GUIDs* in the Windows Installer Editor Help.

   - **Previous Patch GUIDs to replace**
     This lists the GUIDs of previous patches that might have been applied to the original installation. You can browse for additional patches and add their GUIDs to this list. This list of GUIDs should not be delimited; do not enter spaces or other characters between the GUIDs.

     If any of these patches are found on the destination computer and are registered with Windows Installer, they are unregistered and their patch transforms are removed from the list of transforms associated with the application. This lets you apply a patch to an original installation without having applied any intermediate patches. Example: If this patch represents Service Pack 2, setting this field lets end users upgrade to Service Pack 2 even if they did not install Service Pack 1.

   - **Read Product Codes to Upgrade from Target .MSI Files**
     By default, the product codes are read from the .MSI files of the previous versions that you specified. If you do not want the product codes to be read from the previous version .MSI files, clear this check box.

   - **Additional Product Codes**
     If you cleared the previous check box, specify a product code or codes; otherwise no valid targets exist for this patch file. If the previous check box is marked, product codes you enter are added to the product codes that are read from previous version .MSI files. The product codes must be in GUID format, separated by semicolons.

   - **C++ Symbol File Directories (Optional)**
     If your application is written in C++, you can make your patch files smaller by specifying the directories where your Symbol and Object files are stored for the upgrade version of your application.
Adding a Digital Signature to a Patch

Windows Installer 3.0 or later only.

Use the Specify Digital Signature Settings page to add an Authenticode digital signature to a patch file.

Frequently, updating an application requires more privileges than that of a standard user, and only the administrator has sufficient privileges to run the update. This can result in the application needing to be run with administrator privileges.

You can avoid this problem by signing patches that will be run under Windows Vista or later operating system. To do so:

- Make sure that the original installation was digitally signed.
- Add a digital signature to the patch, using the same certificate that was used to sign the original installation.

When the patch is applied, the Windows Vista or later operating system performs the elevation for the application. This means that a standard user can run the update, and does not have to provide administrator authorization to run the application.

Digital signature methods

The file signing tool that is used to digitally sign a file depends on the type of your digital certificate:

- Public/private key pair files
  This method requires a credentials file (.SPC or .CER) and a private key file (.PVK). This method is supported by the signcode.exe tool. For details, search for “Signcode” in the MSDN Library (msdn.microsoft.com/library/).

- Personal Information Exchange file
  This method requires a Personal Information Exchange file (.PFX), which is a container file for the public/private key information. This method is supported by the signtool.exe tool. For details, search for “Signtool” in the MSDN Library (msdn.microsoft.com/library/).

Requirements

- You must have a valid code signing certificate, which you can obtain from a commercial certificate authority such as Verisign. For a list of certificate authorities, search for “Microsoft Root Certificate Program Members” in the MSDN Library (msdn.microsoft.com/library/).
- You must have the signtool.exe or signcode.exe tool on your computer.
- Signtool.exe requires the CAPICOM 2.0 redistributable to be installed and registered on your computer. CAPICOM provides services for digitally signing applications, and is available from the Microsoft Web site.
- The location of signtool.exe or signcode.exe must be specified on the Digital Signature tab in Wise Options in Windows Installer Editor, or they must be available on the system path.

For more information, search for "User Account Control (UAC) Patching" in the MSDN Library (msdn.microsoft.com/library).
**To add a digital signature to a patch**

In the Patch Creation tool, on the Specify Upgrade Version page, mark **Add a Digital Signature to the Patch** and click Next.

See *Creating a Patch File* on page 130.

Complete the Specify Digital Signature Settings page:

- **Web URL**
  Enter your organization’s Web site address.

- **Descriptive Name**
  Enter the name of your application. This name is embedded in your Authenticode certificate to let end users verify the name of the application they are installing.

- **TimeStamp URL**
  Specify the URL you use for your timestamping service. Timestamping lets end users distinguish between a certificate that has expired but was valid when it was used to sign the installation, and a certificate that was used to sign an installation while it was expired. The timestamping service must be available on your computer to build the installation but does not need to be available to the end user running the installation.

- **Certificate options**
  - **Signtool.exe with Personal Information Exchange file**
    Mark this to use signtool.exe and then specify the Personal Information Exchange file (.PFX) to use. This option requires a password.
  - **Signcode.exe with public/private key pair files**
    Mark this to use signcode.exe and then specify the credentials file (.SPC or .CER) that contains your Digital ID, and your private key file (.PVK).

See also:

*Setting Digital Signature Options* in the Windows Installer Editor Help

**Specifying the Patch Sequence**

➤ *Windows Installer 3.0 or later only.*

**To specify the patch sequence**

1. Access the Patch Sequencing dialog box.
   
   See *Creating a Patch File* on page 130.

2. On the Patch Sequencing dialog box, click Add to specify sequencing information.
   
   The Patch Sequence Details dialog box appears.

3. Complete the dialog box and click OK:

   - **Patch Family**
     Specify the patch family that this patch belongs to.

     To sequence this patch after a specific patch, click Browse and select a patch file (.MSP). This populates the **Patch Family** and the **Sequence Within Family** fields. Example: If you browse to a patch that is in Family01 and has a sequence of 10, the current patch will be added to Family01 with a sequence of
20 (sequences are generated in increments of 10). If the selected patch belongs to multiple families, the first family found is used.

- **Sequence Within Family**
Enter a number to specify the order in which this patch should be applied, relative to other patches in this patch family. The default sequence number is the previous sequence plus 10.

- **Product Code**
Typically, you should leave this field blank, which causes the patch to be applied to all targets in the family. If you enter a product code GUID, then sequencing is used only when the patch is applied to the installation defined by that GUID, relative to other patches in the family.

- **Replace previous sequenced patches with this patch**
Mark this to have this patch supersede the updates provided by earlier patches in this patch family. A patch supersedes earlier patches in a patch family when it includes all functionality contained in earlier patches in the family. A small update patch cannot supersede a minor upgrade or major upgrade patch.

4. On the Patch Sequencing dialog box, add more sequencing information if needed. When you finish, click Next to continue the Patch Creation wizard.

See also:

*About Patch Sequencing* on page 129
*Sequencing Patches* and *MsiPatchSequence Table* in the Windows Installer SDK Help

### Specifying Advanced Patch Settings

**To specify advanced patch settings**

1. On the Compile Patch dialog box of Patch Creation, click Advanced to display the Advanced Patch Settings dialog box.

   See *Creating a Patch File* on page 130.

2. Complete the dialog box and click OK:

   - **Do not create file patches, use entire files in patch package**
     Mark this to have the patch file contain entire files instead of only the changed bits of files. Example: Suppose that only five files have changed between version 1.0 and version 1.0.1 of your application. If this check box is cleared, the resulting patch file contains only the changed bits between the five files; if this check box is marked, the patch file contains the five files in their entirety.

   - **Allow Product Codes to differ between the upgrade and prior versions**
     Mark this to upgrade a previous version even if it has a different product code. This global setting overrides the validation settings you specified on the Previous Version Details dialog box.

   - **Allow Version Number to differ between the upgrade and prior versions**
     Mark this to have the patch be able to upgrade a previous version if its version number is different. This global setting overrides the validation settings you specified on the Previous Version Details dialog box. You should always leave this check box marked.
Create a log file
Mark this to create a log file containing details of the patch creation. If an error occurs in the process, refer to this file for information about what caused the error. The log file has the same name you gave to the output .MSP file with the extension .LOG, and is in the same directory.

Specifying Patch Removal Settings

Windows Installer 3.0 or later only.

You can make a patch removable through Add/Remove Programs. This lets end users uninstall a patch without having to uninstall the entire application. For information on which patches can be removed, see Uninstallable Patches in the Windows Installer SDK Help.

To specify patch removal settings
1. On the Compile Patch dialog box of Patch Creation, click Allow Removal to display the Patch Removal Settings dialog box.
   See Creating a Patch File on page 130.
2. Complete the dialog box and click OK:
   ■ Make This Patch Removable
   Mark this to make this patch removable through Add/Remove Programs.
   When you make a patch removable, enter values for the following meta data, which is used by Add/Remove programs. Only the Classification is required.
   ■ Description
   Enter a brief description of the patch that will appear in Add/Remove programs.
   ■ DisplayName
   Enter a name for the patch that will appear in Add/Remove Programs.
   ■ Classification
   (Required.) Enter text to describe the category of updates that this patch belongs to. The category descriptions are arbitrary. Examples: hot fix, security rollup, update, critical update, service pack, update rollup, and so on.
   ■ ManufacturerName
   Enter the manufacturer or publisher of the application.
   ■ TargetProductName
   Enter the name of the application that this patch updates.
   ■ MoreInfoURL
   Enter a URL where end users can get online support for the application.

UpgradeSync

Using UpgradeSync is one of the steps in preparing software for updates.
See Preparing for Software Updates in the Windows Installer Editor Help.
UpgradeSync compares the current package with the previous version of the package, and does the following to prepare the current package for a patch or upgrade:
Wise Package Studio Tools

- Changes the PackageCode, ProductCode, and ProductVersion properties if necessary.
- Aligns component GUIDs. If GUIDs or key paths for the same component don’t match between the new and old .MSI, the component could inadvertently get deleted because Windows Installer does not recognize the components as being the same. Aligning component GUIDs for matching components prevents upgrades from deleting necessary files in the newer version.
- Detects errors that could cause problems with a patch or upgrade and, if possible, fixes them.

**Note**
UpgradeSync does not create an upgrade or patch; it eliminates the most commonly encountered problems that cause patches and upgrades to work incorrectly. Use UpgradeSync before you create a patch or an upgrade. Use Patch Creation to create a patch and the Upgrades page in Windows Installer Editor to create an upgrade.

For information on the different types of updates and when to change the ProductCode and Product Version, see *Patching and Upgrades* in the Windows Installer SDK Help. UpgradeSync changes your current installation according to Microsoft’s recommendations, based on the type of upgrade you plan to make.

When you add new resources to an upgrade installation, you can use component rules to ensure that the component GUIDs are aligned with those in previous installations.

See *Using Component Rules to Align GUIDs in an Upgrade* in the Windows Installer Editor Help.

See also:
*About GUIDs* in the Windows Installer Editor Help

### Using UpgradeSync

**To use UpgradeSync**

1. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with UpgradeSync.
   - On the Tools tab, double-click UpgradeSync.

2. If the Specify Target Installation File dialog box appears, specify the current version of the package file (.WSI or .MSI) to create a patch or upgrade for. Then click Next.

   The Welcome dialog box appears.

3. In **Previous MSI path**, specify the path of the previous .MSI.
   - You must specify an .MSI here, even if you deployed the package as a single-file executable. The .MSI is created in the same directory as the .EXE during compile.

4. Click Next.

   The Upgrade Type dialog box appears.

5. Select the type of upgrade to create.
- **Small Update**  
  Select this to ship this package as a patch or reinstall. Small updates generally contain minimal updates such as changes to the contents of files. This option changes the package code.

- **Minor Upgrade**  
  Select this to ship this package as a patch or reinstall. Minor upgrades generally contain changes such as new or removed features, files, or other items. This option changes the package code and forces you to increment the product version of the current package if it is the same as the previous .MSI’s product version.

- **Major Upgrade**  
  Select this to ship the current package as an upgrade. You usually ship a package as an upgrade when it contains comprehensive changes, but you can create an upgrade for minimal changes. This option changes the package code and product code, and forces you to increment the product version of the current package if it is the same as the previous .MSI’s product version.

- **Current Version**  
  If you selected either the Minor or Major Upgrade options, and is the same as Previous Version, then you must increment it. If you do not increment it, an error occurs when you click Next.

6. Click Next.

The Upgrading/Patching Issues dialog box appears.

This dialog box lists issues in the installation package that might cause problems when creating an upgrade or a patch. These errors are the most common causes of patch and upgrade failures reported by Windows Installer users. Errors that can be fixed automatically have a check box. The following types of errors cannot be fixed automatically and therefore have no check boxes:

- **File `filename.txt` is a new resource that needs to be added to a new component and assigned to a new subfeature**  
  This means that a new resource has been added to an existing component. This is against Microsoft Windows Installer guidelines and can cause problems. Use Windows Installer Editor to put the resource in its own component in Setup Editor > Component tab.

- **Component `componentname.exe` exists in the previous version and the keypath does not exist in the current install. The component’s contents will be deleted during an upgrade.**  
  This means that the component is in the old version, but not in the new version. This might be intentional, so it is not fixed automatically. If it is not intentional, add the component to the new version to prevent it from being deleted during upgrading or patching. Use Windows Installer Editor to add the component in Setup Editor > Component tab.

7. Mark the check boxes of the errors to fix.

8. Click Save to File to save the errors to a text file.

9. Click Finish.

The necessary changes are made to your package. If you plan to create a patch or upgrade, use Patch Creation or the Upgrades page in Windows Installer Editor.

See **Patch Creation** on page 128 or **Upgrades** in the Windows Installer Editor Help.
Web Capture Conversion

Not available in Standard Edition.

When you use Wise Web Capture, the file that results from the capture is an encrypted .MSI, with the extension .MSI_. You cannot open or install this encrypted file, but you can use Web Capture Conversion to decrypt it.

The computer on which you decrypt the file must have Wise Package Studio installed.

To decrypt the captured file

1. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with Web Capture Conversion.
   - On the Tools tab, double-click Web Capture Conversion.
2. On the Select File to Convert dialog box, specify the .MSI_ file and click OK.

The file is decrypted and the extension is renamed to .MSI.

You also can decrypt the .MSI_ file by double-clicking it in Windows Explorer.

See Capturing With Wise Web Capture on page 245.

Wise Task Manager

Not available in Standard Edition.

Wise Task Manager manages the following operations:
- Importing packages into the Software Manager database.
- Adding, removing, or replacing merge modules that are part of .MSI or .WSI packages in the Software Manager database.
- Compiling .MSI or .WSI packages in Software Manager or remotely compiling packages in Windows Installer Editor.
- Checking packages back into Revision Control in Software Manager.

Each of these managed operations consists of one or more tasks.

Because Wise Task Manager manages these operations, the user who runs them can proceed with other tasks in Wise Package Studio while these operations are processed. If the user is working on a client computer, they can also perform these operations (except for checking packages back into Revision Control) on the Wise Package Studio server, which frees the local computer and improves the performance of the operation.

See Performing Server-Side Operations on page 143.

When a user runs one of the managed operations listed above, Wise Task Manager queues them and executes them in the order in which they are received. If an operation is performed locally, it is performed after any other operations that are in the queue for that computer. If an operation is performed on the server, it is performed after any other operations that are in the queue for the server.

Use Wise Task Manager to:
Cancel the tasks of managed operations. You can cancel only the tasks of operations that you run.

View a task's log file to resolve problems if the task fails.

View information about a task, including:
- Its status.
- Whether the task will be performed locally or on the Wise Package Studio server.
- What computer ran the task.
- How many tasks, if any, are ahead of it and waiting to be executed.
- Other task details.

### Using Wise Task Manager

> Not available in Standard Edition.

**Keys to using Wise Task Manager**

- The tasks for each operation form a group, and every other group of tasks is shaded to distinguish the groups.
- A task can have one of six statuses: Canceled, Completed, Executing, Failed, Waiting, or Warning.
- A Warning status means the task completed, but an error message was written to the task’s log file.
- If a task fails, Wise Task Manager moves to the next task.
- When a task is executing:
  - An icon appears in the status column.
  - Details on the progress of the task appear at the bottom of the Wise Task Manager dialog box unless you select another task.

**To use Wise Task Manager**

1. Do one of the following:
   - On the Tools tab, double-click Wise Task Manager.
   - On the Projects tab, click the Run link to the right of the task or tool associated with Wise Task Manager.
   - In Software Manager or Windows Installer Editor, run a managed operation. See *Wise Task Manager* on page 141.

   The Wise Task Manager dialog box appears. If Wise Task Manager opened during a managed operation, the dialog box remains open only until the operation is completed.

2. To view a task’s details, select the task and click Details.

   The Task Details dialog box appears with the following tabs:
   - **General**
     Displays additional details about the task.
Log File
Displays the log file’s location and any messages that were generated. If a task fails, use the Log File tab to determine the problem. If a task does not generate a log file, this tab does not appear.

3. To change the tasks that appear, click Options on the Wise Task Manager dialog box. By default, Wise Task Manager displays only the tasks belonging to operations you run that do not have a status of Completed.

The Wise Task Manager Options dialog box appears.

a. Complete the dialog box:

- **Show tasks for all users**
  Mark this to display the tasks for all managed operations run by all users who are using the same share point directory.

- **Show tasks from the last**
  Select the time period of the tasks to display.

- **Show only the tasks with the following statuses**
  Mark the statuses of the tasks to display.

b. Click OK.

4. To cancel a task, select the task and click Cancel Tasks.

You can cancel only tasks belonging to operations you ran that have a status of Waiting or Executing. If you cancel one task in an operation, then all the tasks of the operation are cancelled.

### Performing Server-Side Operations

*Not available in Standard Edition.*

Wise Package Studio has a server-side service that lets users on client computers perform certain operations on the Wise Package Studio server. By processing operations on the server, you reduce the workload of the client computer and, if the packages and databases reside on the server, you improve the operation’s performance.

With Enterprise Management Server, Security Setup determines whether you have access to this server-side service.

**Note**
The server can process only one task at a time even if the server has multiple processors.

**Server-side service user account**
The server-side service requires a user account to access the information it needs to perform these server-side operations. This user account is set up when Wise Package Studio is installed on the server. If the password for the user account changes, then this service will not work until the user’s password is updated in the Wise Repository Manager on the server. For information on the Wise Repository Manager, see *Managing the Wise Software Repository* in the *Getting Started Guide*. When this service does not work, the tasks for server-side operations do not appear in Wise Task Manager or their status does not change from Waiting.

See *Wise Task Manager* on page 141.
Adding Files From the Wise Software Repository

*Not available in Standard Edition.*

The Files in Repository dialog box appears when a file that is used by a package in the Wise Software Repository is added to an installation. It might appear when you use SetupCapture, Legacy Setup Conversion, or ApplicationWatch. The Files in Repository dialog box lets you add the version of the file that is in the repository, which ensures that you use the correct versions of file resources in applications you develop.

Example: An approved version of the file Sample.dll is stored in the Wise Software Repository and is used by several packages in the Software Manager database. When you add Sample.dll to an installation, you can select the version in the repository as the source for the file.

When you capture a large application with many files, the repository check can slow SetupCapture considerably. Therefore, you can disable this feature in SetupCapture. Clear the **Enable checking of files against Wise Software Repository** check box in the General Settings of SetupCapture Configuration.

**To add a file from the Wise Software Repository**

1. Do one of the following to display the Files in Repository dialog box:
   - Add files to an installation so that the dialog box appears automatically.
   - In Windows Installer Editor, select Tools > Check Repository Files.

2. Mark the check boxes of files to add and click OK.

The file's source path is set to the same location as the version in the Wise Software Repository. You can see the source path on the File Details dialog box > General tab.

**To hide this dialog box in the future**

From **Show this Dialog**, select Hide. This turns the dialog box off for all instances in which it would normally appear.

To make the dialog box appear again, click the Prompts tab in Wise Options in Windows Installer Editor and activate the dialog box.
Chapter 6
Package Validation

This chapter includes the following topics:

- About Package Validation on page 145
- Validating Installation Packages on page 146
- About Customizing Validation Modules on page 148
- Predefined Validation Modules on page 157
- Windows Vista/Windows 7 Validation on page 158

About Package Validation

Package Validation checks Windows Installer packages for errors based on rules in one or more validation modules. It validates installation files (.MSI and .WSI), merge modules (.MSM and .WSM), and transforms (.MST). When you select a transform, Package Validation checks both the transform and the original .MSI.

When you run Package Validation from the Tools tab, you can select a single package file or you can select a folder that contains multiple files. If you select a folder, Package Validation checks all the .MSI or .MSM files in the folder. It can also check the .MSI or .MSM files in the folder’s subdirectories. When validating multiple packages, we recommend that you run the server-side instance of Package Validation to reduce the time that it takes to perform the validation.

Package Validation contains predefined validation modules. The validation modules are fully customizable to accommodate corporate standards:

- You can select which validation rules to use during the validation test.
- With the Quality Assurance module, you can create new validation modules and new validation rules.

Portions of Package Validation were developed based on public Microsoft documents. It has not been reviewed or certified by Microsoft. This tool helps you prepare a package for certification, but cannot guarantee that the application will meet all Windows Application Specification requirements. For information on the Windows requirements, see Windows Installer and Logo Requirements in the Windows Installer SDK Help.

Note
Package Validation ignores issues pertaining to the group box “enabled” attribute, which the Microsoft validation module falsely reports as an error.

See also:
Validating Installation Packages on page 146
Predefined Validation Modules on page 157
About Customizing Validation Modules on page 148
Windows Vista/Windows 7 Validation on page 158
Validating Installation Packages

Use Package Validation to verify installation packages using predefined or customized validation modules.

If you run Package Validation from the Tools tab, you can select a single package file or you can select a folder. If you select a folder, Package Validation can check all the .MSI or .MSM files in the folder. You also have the option to have it check the .MSI or .MSM files in the folder’s subdirectories.

If you run Package Validation from the Projects tab or from Windows Installer Editor, it checks only the current package.

See About Package Validation on page 145.

To validate installation packages

1. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with Package Validation. The package associated with the current project will be verified. This tool might skip dialog boxes or populate fields based on command-line options defined in Process Templates Setup.
   - On the Tools tab, double-click Package Validation.
   - In Windows Installer Editor, select Tools menu > Package Validation.

2. If the Specify Target Installation File or Folder dialog box appears, select an installation file or a folder that contains multiple files and click Next.

   If you select a single installation file, the file can be an .MSI, .WSI, .MSM, .WSM, or .MST. If you select an .MST and the program cannot find the base .MSI file, you are prompted to specify it.

   If you select a folder that contains multiple installation files, you must also select whether to validate .MSI files or .MSM files. You can also mark Include subdirectories to validate that type of file in the selected folder and its subdirectories.

   The Welcome dialog box appears.

3. To view the description of a validation module, select the module in the list.

4. Mark the validation modules to use to validate the package.

   By default, the list contains predefined validation modules.

   See Predefined Validation Modules on page 157.

   Note
   The tests in Application Specification Logo are a subset of the tests in Internal Consistency; therefore, running both tests at the same time might result in duplicate errors.

5. If this package contains multiple releases, the Release drop-down list appears below the list of validation modules. Select the release to test.

6. To customize or create validation modules, click Customize.

   See About Customizing Validation Modules on page 148.

7. Click Next to start testing.
The Performing Validation dialog box appears.

**Note**
The Windows Vista/Windows 7 Compatibility Checks validation module takes longer to run than others due to the nature of its validation checks.

When the validation is complete, the View / Correct dialog box appears for single installation files and the View dialog box appears for multiple files. The View / Correct dialog box lists validation issues for the selected package. The View dialog box lists the pathname for each package on which validation was performed followed by any validation issues for that package. Validation issues represent areas where a package might not comply with the specifications in the validation modules you selected.

The icon to the left of each issue indicates the issue type.

<table>
<thead>
<tr>
<th>Icon</th>
<th>Issue Type</th>
<th>Indicates</th>
</tr>
</thead>
<tbody>
<tr>
<td>!</td>
<td>Error</td>
<td>A problem that causes incorrect behavior and must be fixed</td>
</tr>
<tr>
<td>?</td>
<td>Warning</td>
<td>A problem that might cause incorrect behavior</td>
</tr>
<tr>
<td></td>
<td></td>
<td>When a validation rule set displays issues, it uses this icon.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See <a href="#">Adding a Validation Rule Set</a> on page 154.</td>
</tr>
<tr>
<td>🇺🇸</td>
<td>Logo Issue</td>
<td>A possible problem based on Microsoft’s logo specifications</td>
</tr>
<tr>
<td>🤖</td>
<td>Wise Package</td>
<td>A possible problem based on checks that are built into Wise Package Studio</td>
</tr>
<tr>
<td>Studio-specific Issue</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

8. Select an issue to view its description in **Details**.

   If the issue was found by a Microsoft validation module, see *ICE Reference* or *Merge Module ICE Reference* in the Windows Installer SDK Help for details on the issue.

9. If the Correct button is enabled when you select an issue, click it to correct the issue.

   When Package Validation checks multiple files, the Correct button does not appear.

**Warning**

If you correct a .WSI or .WSM, it is recompiled to an .MSI or .MSM at the end of validation. If you correct an .MSI or .MSM, errors are not corrected in any corresponding .WSI or .WSM.

If the Correct button is not enabled when you select an issue, the issue cannot be corrected from within Package Validation. Make any necessary changes to the installation package in Windows Installer Editor.
Note
The Correct button is not enabled when an issue is found by a custom action rule. Depending on how the custom action is written, the problem might be fixed automatically when it's found, or you might need to fix the problem manually.

10. To add issues to the Task List and fix them manually, mark Add to Task List.
   This option is only available when run Package Validation from Windows Installer Editor.
   When you click Finish, the issues are added to the Task List in Windows Installer Editor.

11. To obtain a record of the issues, click Save to File or Print All.
   When Package Validation checks multiple files, you can save the issues for a specific file. After you select the file in the list, click Save to File > Only Selected.

12. Click Finish.

About Customizing Validation Modules
A validation module is a .CUB file that contains one or more validation rules. A validation rule can be a custom action that calls a .DLL, .EXE, or VBScript (.VBS), or a rule set that consists of a series of conditions and actions.

You can customize validation modules by selecting which validation rules to use during the validation test.

See Selecting Validation Rules to Use on page 150.

With the Quality Assurance module, you also can create new validation modules and new validation rules. Do this if you prefer not to modify a predefined validation module, or to define a validation module that contains only your custom rules.

With Enterprise Management Server, Security Setup determines whether you can customize validation modules.

Adding a Validation Module to Package Validation
✓ Quality Assurance module only.

Package Validation contains predefined validation modules. If the predefined validation modules do not meet your needs, you can add new ones. These can be validation modules you create in the Customized Validation Rules dialog box or in another program such as the Orca database editor.

To add a validation module to Package Validation
1. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with Package Validation. The package associated with the current project will be verified. This tool might skip dialog boxes or populate fields based on command-line options defined in Process Templates Setup.
   - On the Tools tab, double-click Package Validation.
   - In Windows Installer Editor, select Tools menu > Package Validation.
2. If the Specify Target Installation File or Folder dialog box appears, select an installation file or folder and click Next.

3. Click Customize on the Welcome dialog box.

The Customized Validation Rules dialog box appears. **Validation Files** lists the predefined validation modules and any validation modules you’ve added. When you select a validation module, its rules appear in **Validation Rules**.

4. Click Add to the right of the **Validation Files** list.

The Validation File Details dialog box appears.

5. Enter a unique name and description to identify the validation module in the Welcome dialog box of Package Validation.

6. In **File**, specify a new or existing .CUB file name.

   If this is a new file and you do not specify its location, it is created in the Validation subdirectory of your share point directory.

7. If the .CUB file is intended to validate merge modules only, mark **Only display this file for Merge Modules**.

8. Click OK.

   In the Customized Validation Rules dialog box, the validation module you specified appears in **Validation Files**.

9. To add a rule to the validation module, click Add to the right of the **Validation Rules** list. To add a rule that:

   - Calls a custom action.
     
     See **Adding a Rule That Calls a Custom Action** on page 152.

   - Consists of a series of conditions and actions.
     
     See **Adding a Validation Rule Set** on page 154.

10. Enable or disable the rules to use during validation.

    See **Selecting Validation Rules to Use** on page 150.

11. To delete a validation module or rule, select it and click the Delete button to its right.

    Deleting a validation module from Package Validation does not delete the .CUB file from the computer.

12. When you finish, click OK.

The customizations remain in effect until you change them.

**Selecting Validation Rules to Use**

A validation module typically contains multiple rules that verify a package. You can customize validation modules by selecting which validation rules to use during the validation test.

Example:

Darice.cub, authored by Microsoft, contains more than 90 ICE (Internal Consistency Evaluator) custom actions, or rules, that check a package’s internal database consistency. You can disable any of these rules. For a description of Microsoft's ICE custom actions, see **ICE Reference** in the Windows Installer SDK Help.
Note
When customizing a predefined validation module, customize a copy of the .CUB file to retain the original file.

To select validation rules to use
1. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with Package Validation. The package associated with the current project will be verified. This tool might skip dialog boxes or populate fields based on command-line options defined in Process Templates Setup.
   - On the Tools tab, double-click Package Validation.
   - In Windows Installer Editor, select Tools menu > Package Validation.
2. If the Specify Target Installation File or Folder dialog box appears, select an installation file or folder and click Next.
3. Click Customize on the Welcome dialog box.
   The Customized Validation Rules dialog box appears. Validation Files lists the predefined validation modules and any validation modules you’ve added. When you select a validation module, its rules appear in Validation Rules.
4. In Validation Files, select a validation module.
   (Quality Assurance module only.) If the validation module you want is not listed you can add it.
   See Adding a Validation Module to Package Validation on page 149.
5. In Validation Rules, mark the check boxes for the rules to include.
6. (Quality Assurance module only.) To add a rule to the validation module, click Add to the right of the Validation Rules list. You can add a rule that:
   - Calls a custom action.
     See Adding a Rule That Calls a Custom Action on page 152.
   - Consists of a series of conditions and actions.
     See Adding a Validation Rule Set on page 154.
7. When you finish, click OK.
   The customizations remain in effect until you change them.

About Rules That Call a Custom Action

➤ Quality Assurance module only.

You can add rules that call a custom action to a new or existing validation module (.CUB). The custom action, which must be in the form of a .DLL, .EXE, or VBScript (.VBS), performs specific checks on a package.

See Adding a Rule That Calls a Custom Action on page 152.

Note
You cannot add rules to predefined validation modules.
Example: Suppose you want to check packages for hard-coded references to C:\ or D:\ and replace those references with the Windows Installer directory property INSTALLDIR. To do this, write a VBScript to find and replace the references and then display a message. In Package Validation, you specify a new or existing .CUB file and add a rule that calls the VBScript. When you run Package Validation with the new rule enabled, the VBScript performs the find and replace operations.

Errors found by a custom action rule are displayed in the View / Correct or View dialog box only if the custom action contains a message string. When a validation test finds an error based on a custom action rule, the Correct button on the View / Correct dialog box is not enabled; the error is fixed based on functions in the custom action. For information on formatting message strings, see *ICE Message Guidelines* in the Windows Installer SDK Help.

For information on creating a custom action to perform validation checks, see *Internal Consistency Evaluators - ICEs and Building An ICE* in the Windows Installer SDK Help.

**Adding a Rule That Calls a Custom Action**

*Quality Assurance module only.*

**Note**
When customizing a predefined validation module, customize a copy of the .CUB file to retain the original file.

**To add a rule that calls a custom action**

1. Write a custom action (.DLL, .EXE, or VBScript) to perform validation checks.
2. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with Package Validation. The package associated with the current project will be verified. This tool might skip dialog boxes or populate fields based on command-line options defined in Process Templates Setup.
   - On the Tools tab, double-click Package Validation.
   - In Windows Installer Editor, select Tools menu > Package Validation.
3. If the Specify Target Installation File or Folder dialog box appears, select an installation file or folder and click Next.
4. Click Customize on the Welcome dialog box.

   The Customized Validation Rules dialog box appears. *Validation Files* lists the predefined validation modules and any validation modules you’ve added. When you select a validation module, its rules appear in *Validation Rules*.
5. In *Validation Files*, select a validation module to customize.

*Note*
You cannot add rules to predefined validation modules.

If the validation module you want is not listed, add it.

See *Adding a Validation Module to Package Validation* on page 149.
6. Click Add to the right of the **Validation Rules** list and select DLL Rule, EXE Rule, or VBScript Rule.

   A details dialog box appears.

7. Enter a unique name and description to identify this rule when it appears in the **Validation Rules** list.

8. If the custom action file you want is in the **File** drop-down list, select it.

   The files listed are those contained in the validation module you are editing.

9. If the custom action file you want is not in the **File** drop-down list, do one of the following:

    - For .DLL and .EXE files, click New and specify the .DLL or .EXE file.
    - For a VBScript file, click Options, select New, and specify the VBScript file. To edit the VBScript, click Options, select Edit, and modify the VBScript in the Macro Editor.

10. (.DLL and VBScript only.) If the **Function** field appears, enter the name of the function in the .DLL or VBScript that performs the validation.

11. (.EXE only.) If the **Parameters** field appears, enter any command-line options that are needed to run the .EXE.

12. Click OK.

   The OK button is unavailable if required fields are missing.

   The Customized Validation Rules dialog box reappears, and the new rule is displayed at the end of the **Validation Rules** list with its check box marked.

13. To add more rules to this validation module, either repeat this procedure or see **Adding a Validation Rule Set** on page 154.

   The customizations remain in effect until you change them.

---

### About Validation Rule Sets

> **Quality Assurance module only.**

You can add a rule set to a new or existing validation module. A rule set consists of a series of conditions to check in a package and actions to take if the conditions are met.

You create rules sets with the Validation Rules wizard.

See **Adding a Validation Rule Set**.

---

**Note**

You cannot add rules to predefined validation modules.

---

**Example:**

Suppose you want to determine if your packages contain any version of report.dll earlier than 2.0.0 and if so, replace it with version 2.0.0 of report.dll. The rule set you create for this check would contain the following conditions and actions:

- **Select file with name** report.dll
- and where version is **less than 2.0.0**
- **Display text** report.dll version less than 2.0.0 in View/Correct or View dialog boxes
- **Replace file with** C:\Development\DLLs\report.dll

---

Wise Package Studio Reference 152
The underlined values in the conditions and actions might be truncated on the screen. When you include the action to display text, errors found by a custom rule set are displayed in the View / Correct or View dialog box. The error text is displayed with the question mark icon (elope). When you validate a single file, the Correct button is enabled when you select the error. You can then click Correct to perform the actions in the rule set.

**Adding a Validation Rule Set**

> **Quality Assurance module only.**

**Note**
You cannot add rules to predefined validation modules.

**To add a validation rule set**

1. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with Package Validation. The package associated with the current project will be verified. This tool might skip dialog boxes or populate fields based on command-line options defined in Process Templates Setup.
   - On the Tools tab, double-click Package Validation.
   - In Windows Installer Editor, select Tools menu > Package Validation.

2. If the Specify Target Installation File or Folder dialog box appears, select an installation file or folder and click Next.

3. Click Customize on the Welcome dialog box.
   The Customized Validation Rules dialog box appears. **Validation Files** lists the predefined validation modules and any validation modules you’ve added. When you select a validation module, its rules appear in **Validation Rules**.

4. In **Validation Files**, select a validation module to customize.
   If the validation module you want is not listed, add it.
   See *Adding a Validation Module to Package Validation* on page 149.

5. Click Add to the right of the **Validation Rules** list and select Rule Set.
   The Validation Rules dialog box appears.

6. Enter a unique rule set name and description to identify this rule in the **Validation Rules** list.

7. Click Add.
   The Validation Rules wizard starts and the Name dialog box appears.

8. Complete the dialog box and click Next:
   - **Rule name**
     Enter a unique name for this rule.
   - **Type**
     Select whether an issue that breaks this rule will be displayed as an error or warning in the View / Correct or View dialog boxes.
The Conditions dialog box appears.

9. **In Which condition(s) do you want to check**, mark conditions in the order they should be checked.

   The conditions you mark appear in the Rule description list.

10. If a condition contains underlined text, click the underlined text to open the Rule Details dialog box and specify its value.

   Example: If you selected the condition:
   
   Select file with name any
   
   you would click the word any and enter a specific file name.

11. When you finish adding conditions, click Next.

   The Actions dialog box appears.

12. **In Which action(s) do you want to perform**, mark the action or actions to perform if the conditions for this rule are met. Mark them in the order they should be performed. Actions that are incompatible with the conditions you selected are unavailable.

   The actions you mark appear in the Rule description list.

   **Note**
   
   To display a message in the View / Correct or View dialog boxes in Package Validation, include the action Display text [text] in View/Correct or View dialog boxes. Replace [text] with your message.

13. If an action contains underlined text, click the underlined text to open the Rule Details dialog box and specify its value.

   Example: If you selected the action:
   
   Display text none in View/Correct or View dialog boxes
   
   you would click the word none and enter specific text.

14. When you finish adding actions, click Finish.

   The Validation Rules dialog box reappears. The rule list displays the rule, and the Rule description list displays its conditions and actions.

15. To add additional rules, click Add to restart the Validation Rules wizard.

16. Modify rules in the Validation Rules dialog box as follows:

   - To edit a rule, select it in the rules list and click Details. This restarts the Validation Rules wizard, in which you can change the rule name or any conditions or actions.
   
   - To delete a rule, select it in the rules list and click Delete.
   
   - To rearrange the order, click Move Up or Move Down.
   
   - To change the value in a condition or action, click its underlined text in the Rule description list and enter new text in the Rule Details dialog box.

17. When you finish adding and editing rules, click OK in the Validation Rules dialog box.

   The OK button is unavailable if required fields are missing.
The Customized Validation Rules dialog box reappears, and the new rule set is displayed at the end of the Validation Rules list with its check box marked.

18. To add more rules to this validation module, either repeat this procedure or see Adding a Rule That Calls a Custom Action on page 152.

**Editing a Predefined Validation Rule**

➤ Quality Assurance module only.

**Note**

When customizing a predefined validation module, customize a copy of the .CUB file to retain the original file.

You can edit an ICE (Internal Consistency Evaluator) validation rule in a predefined validation module, and select a different custom action (.DLL, .EXE, or VBScript) to perform validation checks for that rule.

**To edit a predefined validation rule**

1. Do one of the following:

   • On the Projects tab, click the Run link to the right of the task or tool associated with Package Validation. The package associated with the current project will be verified. This tool might skip dialog boxes or populate fields based on command-line options defined in Process Templates Setup.
   
   • On the Tools tab, double-click Package Validation.
   
   • In Windows Installer Editor, select Tools menu > Package Validation.

2. If the Specify Target Installation File or Folder dialog box appears, select an installation file or folder and click Next.

3. Click Customize on the Welcome dialog box.

   The Customized Validation Rules dialog box appears. Validation Files lists the predefined validation modules and any validation modules you’ve added. When you select a validation module, its rules appear in Validation Rules.

4. In Validation Files, select a validation module.

   If the validation module you want is not listed, add it.

   See Adding a Validation Module to Package Validation on page 149.

5. In Validation Rules, select a rule and click Details.

   The Details dialog box appears.

6. On the Details dialog box that appears, you can:

   • Edit information about the rule (name, description, function).
   
   • From File, select a custom action to call, or click New to browse for a custom action.

7. Click OK.
Predefined Validation Modules

Package Validation contains predefined validation modules that perform the tests described below.

- **Windows 2000 Application Specification Logo**
  Runs logo.cub, which is provided by Microsoft as part of its Windows 2000 logo verification program. The tests in logo.cub are a subset of the tests in darice.cub, therefore, running both tests at the same time might result in duplicate errors.

- **Windows XP Application Specification Logo**
  Runs XPlogo.cub, which is provided by Microsoft as part of its Windows XP logo verification program. The tests in XPlogo.cub are a subset of the tests in darice.cub, therefore, running both tests at the same time might result in duplicate errors.

- **Windows Installer SDK Internal Consistency**
  Runs darice.cub, which is provided by Microsoft. It performs more than 50 checks to ensure that the installation databases are internally consistent.

- **Merge Module Consistency Checks**
  Runs mergemod.cub, which is provided by Microsoft to test the internal consistency of merge modules. This is the only predefined validation module that appears on the Welcome page when you validate a merge module.

- **Windows Vista/Windows 7 Compatibility Checks**
  Runs WiseVistaIce.cub, which checks for adherence to Microsoft’s package validation requirements for the Vista and later operating systems.

Wise Package Studio Checks

The following checks are built in to Package Validation; they do not run .CUB files.

- **Component design**
  Checks that the proper files have been placed into each component and that the same file has not been placed into multiple components.

- **Uninstall support**
  Checks that the package does not reference non-Windows Installer uninstall utilities.

- **Files in shared folders**
  Checks that no executable files have been placed in shared directories.

- **Hidden and protected application files**
  Checks that application files have been protected from accidental deletion by the end user.

- **Transitive flag set on OS based components**
  Checks that the Transitive bit is set on operating system-dependent components. Transitive components are components that must be swapped out if the end user upgrades the destination computer to a new operating system.

- **Icon placement in Start Menu**
  Checks that shortcut icons are installed correctly in the Start Menu.

- **All application/library files are 32-bit**
  Checks that no 16-bit components are included in the package.

- **Changes to Win.ini or System.ini**
  Checks to see if changes are being made to these files.
Package Validation

- **Components shared with non-Windows installer applications**
  Checks that no components of the package are shared with other applications that do not use Windows Installer.

- **Files installed to Program Files by default**
  Checks that all application files are installed to a subdirectory of Program Files.

- **Terminal Server Compatibility**
  Checks for errors that might cause problems when the package is installed in a Microsoft Terminal Services or Citrix environment. With terminal service applications, installation resources must reside in per-machine locations rather than per-user locations. Errors result from this test if the installation is set to install per-user, if any keypaths reside in user-specific locations, or if environment variables are present in the installation. If environment variables are present, using the Correct button duplicates the variables, creating a per-user set and a per-machine set, one of which is installed depending on the value of ALLUSERS. Correcting some errors might cause keypaths to be empty, and might cause a one-time repair. You can set a release to be compatible with Terminal Services. See `ALLUSERS Property` in the Windows Installer SDK Help.

  Also see the description of the **Release Type** field in `Creating a New Release` in the Windows Installer Editor Help.

See also:

- `ICE Reference` in the Windows Installer SDK Help
- `Merge Module ICE Reference` in the Windows Installer SDK Help

### Windows Vista/Windows 7 Validation

The Windows Vista/Windows 7 Compatibility Checks validation module in Package Validation runs WiseVistaIce.cub, which checks for adherence to Microsoft’s package validation requirements for the Vista and later operating systems. For information about Windows Vista logo requirements, visit the MSDN Library (msdn.microsoft.com).

**Updating the list of protected files and registry keys**

The Windows Vista/Windows 7 validation verifies that the installation does not contain files or registry keys that are protected by Windows Resource Protection (WRP). To perform this check, Package Validation compares resources in the installation to those in the following tables in WiseVistaIce.cub: `_VistaProtectedFiles` and `_VistaProtectedRegKeys`. The contents of these tables are current as of this product’s release. You can update these tables as follows:

- Edit WiseVistaIce.cub directly.

- Rebuild the tables by running the GetWRPItems.exe utility, which is installed in the bin subdirectory of this product’s installation directory. Run it from the command prompt on a computer that is running Windows Vista or later operating system and specify the full path to WiseVistaIce.cub. Example:

  "C:\Program Files\Symantec\Wise Package Studio\bin\getwrpitems.exe" "C:\Wise Share Point\Validation\wisevistaice.cub"

  The GetWRPItems.exe utility deletes the contents of both tables and rebuilds them based on the WRP information on your computer. Because the edition of the Windows Vista or later operating system that is installed might affect the contents of
the rebuilt files, run the utility on the edition of the operating system that your installations target.
This chapter includes the following topics:

- **About Test Expert** on page 159
- **Opening a Package in Test Expert** on page 160
- **About Test Cases** on page 164
- **Test Case Reference** on page 172
- **Installation Tests** on page 173
- **Standard Tests** on page 177
- **Application Verification Tests** on page 180
- **Application Execution Tests** on page 187
- **Uninstall Tests** on page 193

### About Test Expert

*Quality Assurance module only.*

Test Expert provides a structured, intelligent approach to testing Windows Installer packages that eliminates the random approach often used in an ad hoc testing environment. Test Expert generates test cases based on the .MSI contents and lets you add test cases that fit your organization's needs. Testers are guided step-by-step through test plans and can view the test status at any stage of the testing. You can test .MSIs, .EXEs that were compiled from .MSIs, and groups of packages from the Wise Software Repository.

Use Test Expert as part of your organization's quality assurance process for testing Windows Installer packages. Test Expert generates test cases, lets you create your own test cases, tracks statuses, and helps you create a test plan that fits your organization's needs.

Test Expert reads package contents and generates a Master Test Plan that contains test cases based on the contents. Example: If an .MSI contains launch conditions that require Windows XP and Windows Vista, tests cases are created to test each launch condition.

Generated tests are run within Test Expert. The buttons in the Test Expert toolbar change to let you run each type of test. (Example: If you are running an installation test, the Install button appears and clicking it runs the installation.) Some tests run automatically, some tests start a process but wait for you to visually confirm results, and some tests must be run manually and might require varying degrees of manual configuration.

When you run an installation test, you can install the package into a virtual software layer.

See *Installing an Installation Test into a Virtual Software Layer* on page 164.
If you create your own test cases, you must run them manually outside Test Expert.

**Opening a Package in Test Expert**

➤ **Quality Assurance module only.**

To test a Windows Installer package or group of packages, open it in Test Expert. Ideally, testing is done on a clean machine with all other applications closed to prevent interference with test results. You can apply transforms to a package when you open it.

**To open a package in Test Expert**

1. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with Test Expert. The default project file is opened. This tool might skip dialog boxes or populate fields based on command-line options defined in Process Templates Setup.
   - On the Tools tab, double-click Test Expert.

2. In the Open Application/Package dialog box that appears, choose what to open:
   - **Windows Installer Package or Software Manager Group**
     Mark this to specify an .MSI file, a group in Software Manager, or an .EXE that was compiled from a Windows Installer project. Opening an .EXE that was not compiled from Windows Installer causes an error.
     See [About Testing Groups of Packages](#) on page 166.
   - **Application That Is Installed on This PC**
     Mark this to specify the cached .MSI of an installed application. Every time a Windows Installer installation is run, a randomly-named copy of the .MSI is cached in the system directory. The cached copy contains the installation information and logic, but it does not contain the installation files. Therefore, if you open a cached .MSI, installation tests and uninstall tests do not appear in Test Expert.

3. Click the **...** button to:
   - Test an .MSI or an .EXE that was compiled from an .MSI. Use the File System tab, which shows your local file system.
   - Test a package from the repository. Use the Repository tab—the fields at the bottom serve as filters to narrow your search. Only "distributable" packages appear on the Repository tab, meaning the repository contains the actual resources of the packages, not just information about those resources.
   - Test a group of packages from the repository. Use the Groups tab, which show groups in the repository specified in Database. A group can contain WiseScript-based packages.

   For details on the Open dialog box, see *Opening an Installation Package* in the Windows Installer Editor Help.

4. If you select the **Windows Installer Package or Software Manager Group** option, and you select single package file (.MSI), the Transforms button is enabled. To apply transforms to this package:
   a. Click Transforms.
The Select Transforms dialog box appears.

b. Click Add and specify the transform.

c. Specify additional transforms if necessary.

d. The transforms are applied to the package in the order they appear in the list.
   To rearrange the order, select a transform and click Move Up or Move Down.

5. Click OK in the Open Application/Package dialog box.

The Master Test Plan for the item you opened is displayed. The Master Test Plan contains test cases based on the .MSI contents and any transforms that you applied. At this point, you can add your own test cases or start to execute test cases. Uninstall tests do not appear when groups are open.

See also:

About Test Expert on page 159
Loading, Saving, and Clearing Results Files on page 163

Setting Test Expert Preferences

➢ Quality Assurance module only.

To set Test Expert preferences
1. In Test Expert, select Edit menu > Preferences.
2. On the Prompts tab, reactivate prompts that you previously suppressed. Example:
   If an alert dialog box had a check box labeled Don’t show this message again,
   and you marked it, the prompt would appear here.
   To reactivate a prompt, select it and click Activate.

About the Master Test Plan

➢ Quality Assurance module only.

Each time you open a package or group in Test Expert, test cases are generated based on contents, and the Master Test Plan for that package or group is displayed.

Test groups
Test cases are organized into test groups in the left pane. Each test case can be comprised of one or more test items, which appear in the right pane. Example: A test case for shortcuts might consist of several individual test items, one for each shortcut in the application installation.
Right pane of Test Expert

Opens the test plan for a Windows Installer package or group.

Master Test Plan. Click Test Plan Details to display the master test plan in the right pane.

Test Groups. Only groups that contain test cases are shown.

Test Cases.

Master Test Plan view

The Master Test Plan view in the right pane provides a visual summary of the entire test plan. You do not run tests from the Master Test Plan view; you must first select a test case.

See Running a Test Case on page 165 and About Test Cases on page 164.
Loading, Saving, and Clearing Results Files

Quality Assurance module only.

When you open a package or group in Test Expert, a results file is created to store testing statuses, testing comments, and user-defined test cases. The results file has the extension .WTE.

You can have different results files for different testing regimens. Example: Suppose you are doing extensive testing on Windows XP/2003/Vista. You can maintain a different results file for each operating system, because some tests might pass on one operating system but fail on another.

The results file can be opened and saved from within Test Expert so you can share testing results with co-workers.

- To open a results file, select File menu > Load Results, navigate to a .WTE file, and open it. Any current testing results are immediately deleted.
- To save the current results file, select File menu > Save Results, name the file, and save it.
- To clear the current results file, select File menu > Clear Results. The status of all test cases is changed to Pending and all comments associated with tests are deleted.
Installing an Installation Test into a Virtual Software Layer

Requires Software Virtualization Agent.

When you run an installation test, you can install the package into a virtual software layer. After you finish testing the package, you can use the Symantec SVS applet to delete or deactivate the virtual software layer. If you run the uninstall tests, the layer is deleted.

This has the following benefits:

- It restores the computer to its original state so that you do not need to reimage it to test another package.
- If you created a snapshot when you ran the installation test, you can use this snapshot when you run the next installation test.

When you run an installation test and install the installation into a virtual software layer, all activated layers are deactivated. If the installation test depends on something in a virtual software layer, it will not be available because the layer is deactivated. When the installation is complete, the layers that were deactivated are reactivated.

When you run an uninstall test on a package that was installed into a virtual software layer, the layer must be activated.

About the Altiris SVS Applet in the Virtual Package Editor Help.

About Test Cases

Quality Assurance module only.

Different kinds of test cases are run differently. (Example: Generated tests are run within Test Expert, but user-defined tests must be run manually outside Test Expert.) Some tests might have to be run on multiple computers.

When you select a test case in the left pane, the upper-right pane shows information for the selected test case:

- Description of the test case
- Statistics for test items that make up the test case
- Overall status of the test case.

Use the Pending, Passed, or Failed tabs to view test items by status.
Running a Test Case

> Quality Assurance module only.

This procedure contains general guidelines for running generated test cases. Each type of test case is more fully explained in its own topic. Running a test case if you have a group open is slightly different.

See About Testing Groups of Packages.

To run a test case

1. Prepare the testing computer. Depending on what you are testing, this might include:
   - Installing the application.
   - Uninstalling the application.
   - Setting display or other options to match or fail launch conditions.
2. Select a test case.
3. If a Run column with check boxes appears in the lower-right pane, mark the test items to run. All check boxes are marked by default.
4. To start the test, click one of the buttons at the top of the window: Install, Uninstall, Execute, or Run. Not all buttons appear for all tests; only relevant buttons appear. For a description of each test case, see Test Case Reference on page 172.
5. After test execution, set the status of test items to Pass or Fail. Some tests set this automatically. Then set the overall status of the test case to Pass or Fail.

See Setting Test Statuses and Details on page 166.
About Testing Groups of Packages

Running a test case for a Software Manager group is slightly different from running a test case for a single package.

- When a group is open, you see a **Package** drop-down list at the top of Test Expert. This lists the packages in the group. It does not list the transforms because they are already applied. Use this list to view the test cases for any one package in the group of all packages in the group.

- If you have a group open, the Package Execution dialog box opens for installation tests, from which you install each package in the group.

- Uninstall tests do not appear for groups.

- If transforms are included in a group, Test Expert opens the base .MSI package as though the transforms are already applied.

Setting Test Statuses and Details

➤ **Quality Assurance module only.**

You can set the status for a test case or a test item. Test cases are listed in the left pane of Test Expert and test items appear in the right pane after you select a test case name.

- Some tests, particularly application verification tests, are run automatically and set test item statuses automatically. In some cases, the overall status of the test case is also set automatically. In other cases, you must set the overall status of the test case.

- Some tests require visual verification from you and prompt you to set the status.

- Some tests are informational and might not require a Pass or Fail status for each test item. Example: When you run the Extra Files or Extra Registry Entries test cases, you see a list of items that were accessed but not installed. These items do not necessarily require a Pass or Fail status, but the Status column is available if you need to mark an item as a problem.
Test case and test item statuses in Test Expert

**Status of Test Case.** Set the overall status of the selected test case, which, in this example, is **File Extensions**.

**Status of Test Cases.** This shows the overall status for each test case. The absence of an icon indicates the test is Pending.

**Status of Test Items.** This column shows the status for the test items that make up a test case. Right-click one or more test items and select Set Status to change status for test items. Some statuses are set automatically.

See *About Test Cases* on page 164 for explanations of other elements of the Test Expert user interface.

**To set the status of a test case**

1. Select the test case name in the left pane. The test case information is displayed in the right pane.
2. In the upper-right pane, select a status from **Status of Test Case**.

**To set the status of a test item**

- Right-click on a test item in the lower-left pane. Select Set Status and then select the status. You can select multiple test items before right-clicking.

**OR**

- For semi-automated tests, such as File Extensions, select the status in the Pass/Fail dialog box that appears after you visually verify the test.
  
  See *Application Verification Tests* on page 180.

**OR**

- For automated tests, such as Prog IDs, the status is set automatically.

**To change details of a test item**

- Double-click a test item and add comments in the **Test Details** field of the dialog box that appears. This field is replaced each time a test is run.
Determining Your Test Environment

➤ Quality Assurance module only.

You can use Test Expert to run tests on a clean machine, a baseline machine, or both. The test environment you choose depends on your organization’s requirements. A clean machine contains only the operating system, service packs, and runtimes of a typical computer in your organization. A baseline machine contains the software packages that are common to all computers in your organization, which replicates the environment where the software will be running. Example: If all computers contain Microsoft Office, a baseline machine should contain Microsoft Office.

You can test a package and then restore a clean or baseline machine to its original state without reimaging it.

See Installing an Installation Test into a Virtual Software Layer on page 164.

Because Wise Package Studio cannot be installed on operating systems earlier than Windows 2000, you cannot use Test Expert on these operating systems. If your target computers include these operating systems, use another method for testing your installations on these systems.

Testing on Multiple Computers

➤ Quality Assurance module only.

Some test cases in Test Expert might have to be run on multiple computers. (Example: To test a launch condition that specifies Windows XP, you must install not only on a computer running Windows XP, but also on a non-Windows XP computer.) To easily switch computers, while providing easy access to test case information in Test Expert, you can perform a Wise Package Studio Client installation.


Process for testing on multiple computers

**Wise Package Studio Server**
- Contains the complete version of Wise Package Studio and points to a share point directory.
- Packages are saved in share point directory

**Wise Package Studio Clients** (which point to the share point directory)
On each computer:
- Start Test Expert and open a package from the share point directory.
- Run tests.
- Enter results, which are saved in the share point directory.
To test on multiple computers

1. Make sure the package to be tested is in the share point directory.
2. Perform a Server installation of Wise Package Studio on a server.
3. Perform a Client installation of Wise Package Studio Client on the testing computers.
   This installs only the shortcuts and support files needed to run Wise Package Studio.
   The program files are not installed locally, but are run from the packaging server.
4. On the testing computers, open Test Expert, and open the package.
   The results file that is stored with the .MSI is read and the current testing status of the
   .MSI is displayed in Test Expert.

Note
If you don’t have permission to write to the directory that contains the package, results
files are stored in the temp directory of the local computer.

Machine Capture Settings

- Quality Assurance module only.

When you click Install, Install As, or Uninstall in Test Expert, a wizard appears in which
you can run Machine Capture. Machine Capture creates a snapshot of the current state
of the computer. It provides results for uninstall tests by comparing pre-installation and
post-uninstall snapshots. Uninstall tests are not available if a group is open.

See Uninstall Tests on page 193.

In the initial dialog box of Machine Capture, you can set options that govern how the
snapshot is created. These options are saved in a configuration file that is shared with
both SetupCapture and SOE Snapshot. See:

- Setting Directories to be Watched for Uninstall Tests
- Setting a File, Wildcard, or Directory to Be Ignored During Uninstall Tests on
  page 170
- Setting Registry Entries to be Ignored During Uninstall Tests on page 171

Setting Directories to be Watched for Uninstall Tests

- Quality Assurance module only.

You can specify the directories to be captured when Machine Capture creates a snapshot
of the current state of the computer. Uninstall tests show only the results from the
directories you specify. Uninstall tests are not available if a group is open.

At a minimum, specify the Program Files directory and the Windows or Winnt directory,
including subdirectories. For the most complete record of changes, specify the C:\ root,
including subdirectories. However, watching the entire root might capture changes that
occur during normal operation and have nothing to do with the installation. You can
exclude irrelevant changes to avoid recording them.

See Setting a File, Wildcard, or Directory to Be Ignored During Uninstall Tests on
page 170 and Setting Registry Entries to be Ignored During Uninstall Tests on page 171.
To set directories to be watched for uninstall tests
1. Start Machine Capture by clicking Install or Install As in Test Expert.
   The Welcome dialog box appears.
2. Click Settings.
3. On the Machine Capture Settings dialog box, click the Directories to Watch tab.
4. Add directories to watch:
   a. Click Add.
      The Select Directory dialog box appears.
   b. Click a directory to add it to the list of watched directories. To watch all the
      subdirectories of the directory also, mark **Include Sub-Directories**.
   c. Click OK.

To edit a directory entry, double-click it in the list.

See also:
- Files and Registry Entries Ignored During Captures on page 247
- Uninstall Tests on page 193

Setting a File, Wildcard, or Directory to Be Ignored During Uninstall Tests

➤ Quality Assurance module only.

You can specify files and directories to ignore when Machine Capture creates a snapshot of
the current state of the computer. Uninstall tests omit these files and directories from
uninstall test results. Uninstall tests are not available if a group is open.

Generally, you should exclude anything that is temporary or anything that is likely to
result from applications other than the one you are testing. Example: An installation
might temporarily store files in a temp directory, but those files have no effect on the
installation.

To a file, wildcard, or directory to be ignored during uninstall tests
1. Start Machine Capture by clicking Install or Install As in Test Expert.
   The Welcome dialog box appears.
2. Click Settings.
3. On the Machine Capture Settings dialog box, click the File and Folder Exclusions tab.
   Recommended exclusions are listed. You can edit or replace them.
   See Files and Registry Entries Ignored During Captures on page 247.
4. Click Add.
   The File Exclude dialog box appears.
5. In the File Exclude dialog box:
   a. In **Directory**, specify a directory. To exclude the file or wildcard specified below
      for all local drives, leave this field blank.
b. In **File/Wildcard**, do one of the following:

- Specify a file.
- Specify a wildcard for part of the file name. Enter * to represent any number of characters, or enter ? to represent any single character. You can enter multiple wildcards separated by semicolons.
- To ignore all files in the directory specified above, leave **File/Wildcard** blank.

6. To ignore the subdirectories of the specified directory, mark **Exclude Sub-Directories**.

7. Click OK.

To edit an exclusion, double-click it in the list.

See also:

- Setting Directories to be Watched for Uninstall Tests on page 169
- Setting Registry Entries to be Ignored During Uninstall Tests on page 171
- Uninstall Tests on page 193

### Setting Registry Entries to be Ignored During Uninstall Tests

➤ Quality Assurance module only.

You can specify registry entries to ignore when Machine Capture creates a snapshot of the current state of the computer. Uninstall tests omit these registry entries from uninstall test results. If this value or key changes during the installation, it will be ignored in the results for uninstall tests. Uninstall tests are not available if a group is open.

Generally, you should exclude anything that is temporary or anything that is likely to result from applications other than the one you are testing.

**To set registry entries to be ignored during uninstall tests**

1. Start Machine Capture by clicking Install or Install As in Test Expert.
   
   The Welcome dialog box appears.

2. Click Settings.

3. On the Machine Capture Settings dialog box, click the Registry Exclusions tab.
   
   Recommended exclusions are listed. You can edit or replace them.

   See Files and Registry Entries Ignored During Captures on page 247.

4. Click Add.
   
   The Exclude Registry Key dialog box appears.

5. In the left pane, navigate to and select a registry key.
   
   - To exclude the entire registry key, click `<ignore entire subtree>` in the right pane.
   - To exclude a specific value, click the value name in the right pane.
6. Click OK.

To edit an exclusion, double-click it in the list.

See also:

Setting Directories to be Watched for Uninstall Tests on page 169
Setting a File, Wildcard, or Directory to Be Ignored During Uninstall Tests on page 170
Uninstall Tests on page 193

Adding a User-Defined Test Case

➤ Quality Assurance module only.

You can add your own test cases to the test plan in Test Expert. You cannot add multiple test items under a test case.

User-defined tests must be run manually outside Test Expert. After you run a manual test case, update the status of the test case in Test Expert.

See Setting Test Statuses and Details on page 166.

To add a user-defined test case

1. Right-click in the left pane of Test Expert and select Add Test Case.

   The Add User-Defined Test Case dialog box appears.

2. Complete the dialog box:

   • **Name**
     Enter a unique name for this test case. You cannot use the name of any of the test cases that Test Expert generates.

     See Test Case Reference on page 172.

   • **Group**
     Select the group that matches the type of test case you are adding. The test groups are those that are listed in the left pane of Test Expert.

   • **Description**
     Enter a description of the test case, such as steps on how to execute it.

3. Click OK on the Add User-Defined Test Case dialog box.

   The new test case appears in the test group you specified.

Test Case Reference

➤ Quality Assurance module only

See the following topics for information on each of the test cases that Test Expert generates.

Installation Tests on page 173

Launch Conditions Test Case on page 176

OS Conditions Test Case on page 176

Verify Installation Test Case on page 177
**Installation Tests**

- **Quality Assurance module only.**

  In Test Expert, installation tests are verified by running the installation. The Install and Install As buttons appear only for installation tests, which appear only if the package is not yet installed.

  See *How to Run Installation Tests* on page 174.

  Installation tests and uninstall tests do not appear if you opened a cached copy of an .MSI. Every time a Windows Installer installation is run, a randomly-named copy of the .MSI is cached in the system directory. The cached copy contains the installation information and logic, but it does not contain the installation files.
When you run an installation test, you can install the package into a virtual software layer.

See *Installing an Installation Test into a Virtual Software Layer* on page 164.

During installation tests, you can run Machine Capture to create a pre-installation snapshot of the computer. Later, when the application is uninstalled, you run Machine Capture to create a post-uninstall snapshot. Uninstall tests compare the differences between the two snapshots. Uninstall tests are not available if a group is open.

If you encounter problems while running an Installation Test, run it in the Debugger for Windows Installer. To do this, close Test Expert, open the package in Windows Installer Editor, and click Debug. The Debugger provides detailed runtime information such as property values, lets you edit values, and lets you set breakpoints.

See also:

- *Launch Conditions Test Case* on page 176
- *OS Conditions Test Case* on page 176
- *Verify Installation Test Case* on page 177
- *About Test Cases* on page 164

## How to Run Installation Tests

➤ *Quality Assurance module only.*

### How to run installation tests

1. Close all applications other than Wise Package Studio and Test Expert, including all background applications and services that might access files or registry entries on the testing computer. Files and registry entries accessed by other applications can interfere with the test results.

2. In Test Expert, select a test case under the Installation Tests group in the left pane.

   **Note**
   You simultaneously test all installation tests. Note the conditions specified in Launch Conditions and OS Conditions. While verifying installation, note what test items of Launch Conditions or OS Conditions were verified also, and after installation, mark their statuses.

3. Make sure the applications installed by the package or group are uninstalled, because installation tests install those applications and monitor installation results.

4. Click one of the following buttons. If these buttons do not appear, make sure you have uninstalled the .MSI.

   - **Install**
     Run the installation normally. If you currently have a group open, this is the only button that is available.

   - **Install As**
     Run the installation as a user other than the logged on user. This lets you test the installation with a different privilege level. Later in this process, a dialog box appears in which you specify the user information. (The Install As button does not appear for groups.)
If the Software Virtualization Agent is installed on the computer, the Welcome dialog box of the Installation Test wizard appears. Otherwise, the Machine Capture dialog box appears.

5. If the Welcome dialog box appears, specify whether to install the application into a virtual software layer and click Next.

See *Installing an Installation Test into a Virtual Software Layer* on page 164.

6. If you will use Machine Capture to create a pre-installation snapshot, then click Settings on the Machine Capture dialog box to view or edit the capture settings.

See *Machine Capture Settings* on page 169.

If you intend to run uninstall tests later, then create the pre-installation snapshot, which is used during uninstall tests.

7. On the Machine Capture dialog box, do one of the following:
   - To create a snapshot, mark **Create New Snapshot** and click Next. If this option does not appear, just click Next.
   - To use an existing snapshot, mark **Use Existing Snapshot** if it appears and click Finish.
   - To skip capturing a pre-installation snapshot, click Cancel.

8. If you create a snapshot, the Capturing Machine State dialog box appears and the scan begins, which takes a few moments. When the capture is complete, click Finish.

9. If you have a group open, the Package Execution dialog box opens, from which you install each package in the group. The packages are in the order that they were put in Software Manager.

**Note**

Non-.MSI-based packages might show erroneous information in the Status column.

- **Install**
  Run a single installation normally.

- **Install As**
  Run an installation as a user other than the logged on user.

- **Install All**
  Run all installations normally.

- **Install All As**
  Run all installations as a user other than the logged on user.

10. If you clicked Install As or Install All As, a dialog box appears that lets you specify user logon information. The dialog box differs depending on the operating system. In this dialog box, enter the user name, domain, and password of a user whose privileges the installation will be run under. Then click OK.

    An installation opens.

11. Run the installation and verify that the installation takes place as you expected. If you are working with a group, run each installation.

12. When the test is finished, select the status for each test item based on your visual verification of the test results.

13. Select the overall status of the test case in **Status of Test Case**.
If Test Expert detects launch conditions in a package, the Launch Conditions test case appears.

A launch condition is a requirement that must be met for the installation to begin. It might be a requirement for a particular operating system, for a certain user privilege level, or for a certain property to be true. Launch conditions usually test values of Windows Installer runtime properties, such as properties that specify a required operating system.

To thoroughly test each launch condition:

- Install the package on a computer where each condition is true, and verify that the installation takes place.
- Install the package on a computer where each condition is false, and verify that the installation fails.

Example: If a launch condition requires a specific operating system, install the package on every operating system and verify that it installs successfully only on computers with the required operating system.

It is unlikely that testing on only one computer will adequately test launch conditions. See Testing on Multiple Computers on page 168.

When you finish the test, select the status for all test items and the overall status of the test case based on your observation of the installation.

See Setting Test Statuses and Details on page 166.

See also:
How to Run Installation Tests on page 174
About Test Cases on page 164

If Test Expert detects component conditions in a package, the OS Conditions test case appears. This test case searches the package for all components that have a condition that includes the Windows Installer property names Version9X or VersionNT.

In Windows Installer Editor, you set a component condition by adding a condition to a feature in Installation Expert > Features page. The condition is applied to every component in the feature.

To test these conditions:
Use the Verify Installation test case to install the package on the required operating system.

Run the application execution tests. If the application performs as expected, the component was probably installed correctly. To verify whether a component was installed, open the .MSI, determine its component GUID, and then search the registry for that component GUID. If the component GUID is in the registry, then the component was installed.

When you finish the test, select the status for all test items and the overall status of the test case based on your observation of the installation.

See Setting Test Statuses and Details on page 166.

See also:
How to Run Installation Tests on page 174
About Test Cases on page 164

Verify Installation Test Case

The Verify Installation test case in Test Expert runs the installation on the current computer so you can verify that the package installs correctly. If you are using Test Expert on multiple testing computers, you can use this test case to also test Launch Conditions and OS Conditions on multiple computers.

During this installation, you are prompted to install the package into a virtual software layer if you have the Software Virtualization Agent installed.

You are also prompted to create a pre-installation snapshot of the computer. This snapshot is not necessary to verify any installation tests, but it is necessary to verify uninstall tests. If you intend to run uninstall tests later, then create the snapshot. Uninstall tests are not available if a group is open.

When you finish the test, select the status for all test items and the overall status of the test case based on your observation of the installation.

See Setting Test Statuses and Details on page 166.

See also:
How to Run Installation Tests on page 174
About Test Cases on page 164

Standard Tests

In Test Expert, standard tests perform a variety of tests, including letting you run any program, which allows for complete customization of your test plan. These tests must be visually verified or must have a verification mechanism built into them. Example: If you set a test to run a program, build a message dialog box into the program that indicates
results. Standard tests are stored in the repository and are not available without a repository, such as with the Standard Edition.

**Requirements**

- Each standard test must be set up with relevant parameters. Because standard tests are stored in the repository, they are shared among team members.

See also:

- [Check Internet Connection](#) on page 178
- [Check Network Location](#) on page 178
- [Database Connectivity](#) on page 179
- [Execute Program](#) on page 180

## Check Internet Connection

⚠️ **Quality Assurance module only.**

Use this test case to check internet connections to specific URLs. This test appears for any package or group that you open, along with a default test item.

**Requirements**

- The computer must have a valid Internet connection.

**To set up and run the test case**

1. Select the Check Internet Connection test case in the left pane of Test Expert.
2. Click Add.
   - The Check Internet Connection dialog box appears.
3. Configure the test item:
   b. Mark **Make this a global test** to have this test item appear for every package or group that is subsequently opened in Test Expert. Otherwise it appears only in the test plan of the currently opened package or group. Global tests are stored in the repository and appear for all other users of the same repository.
   c. Click OK to save the test item configuration.
   d. Use the Edit button to change a test item configuration.
   e. Use the Delete button to remove a test item.
4. To run the test item, select its check box and click Execute.

## Check Network Location

⚠️ **Quality Assurance module only.**

Use this test case to check if a specific network location is readable to the computer. This test case appears for any package or group that you open, but it does not have any test items until you add them.
**Requirements**

- The current user profile must have a valid network connection and appropriate permissions.

**To set up and run the test case**

1. Select the Check Network Location test case in the left pane of Test Expert.
2. Click Add.
   
The Check Network Location dialog box appears.
3. Configure the test item:
   a. In **Network**, enter a UNC network location. Example: \\Server\Department\Data.
   b. Mark **Make this a global test** to have this test item appear for every package or group that is subsequently opened in Test Expert. Otherwise it appears only in the test plan of the currently opened package or group. Global tests are stored in the repository and appear for all other users of the same repository.
   c. Click OK to save the test item configuration.
   d. Use the Edit button to change a test item configuration.
   e. Use the Delete button to remove a test item.
4. To run the test item, select its check box and click Execute.

**Database Connectivity**

» **Quality Assurance module only.**

Use this test case to check if a specific database is accessible. This test case appears for any package or group that you open, but it does not have any test items until you add them.

**Requirements**

- The data source you set must be able to connect to its specified database.

**To set up and run the test case**

1. Select the Database Connectivity test case in the left pane of Test Expert.
2. Click Add.
   
The Database Connectivity dialog box appears.
3. Configure the test item:
   a. In **Data Source**, enter an ODBC data source name. Click Browse to browse this computer’s ODBC data sources.
   b. Mark **Make this a global test** to have this test item appear for every package or group that is subsequently opened in Test Expert. Otherwise it appears only in the test plan of the currently opened package or group. Global tests are stored in the repository and appear for all other users of the same repository.
   c. Click OK to save the test item configuration.
d. Use the Edit button to change a test item configuration.
e. Use the Delete button to remove a test item.

4. To run the test item, select its check box and click Execute.

**Execute Program**

*Quality Assurance module only.*

Use this test case to run any executable program. You can use WiseScript Editor, WiseScript Package Editor, or another development environment to create a wide variety of tests in executable form. If you build your own executable, build in a mechanism into the .EXE to indicate whether the test passed or failed. This test case appears for any package or group that you open, but it does not have any test items until you add them.

**To set up and run the test case**

1. Select the Execute Program test case in the left pane of Test Expert.
2. Click Add.
   
   The Execute Program dialog box appears.
3. Configure the test item:
   a. In **.EXE Path**, enter the path to an executable file. If Test Expert will be used on other computers, specify the executable with a UNC path.
   b. In **Command Line**, enter command lines to apply to the .EXE.
   c. Mark **Make this a global test** to have this test item appear for every package or group that is subsequently opened in Test Expert. Otherwise it appears only in the test plan of the currently opened package or group. Global tests are stored in the repository and appear for all other users of the same repository.
   d. Click OK to save the test item configuration.
   e. Use the Edit button to change a test item configuration.
   f. Use the Delete button to remove a test item.
4. To run the test item, select its check box and click Execute.

**Application Verification Tests**

*Quality Assurance module only.*

In Test Expert, application verification tests examine the tables of a package and then verify data and settings based on the contents. In some cases these tests are run automatically, and in other cases, they require your visual verification of the test's success.

**Requirements**

- The package must be installed on the testing computer.
- The package must contain an .EXE that runs the application.
See also:

Class IDs Test Case on page 181
File Extensions Test Case on page 182
Help Files Test Case on page 183
ODBC Data Sources Test Case on page 184
Prog IDs Test Case on page 184
Search Locations Test Case on page 185
Services Test Case on page 186
Shortcuts Test Case on page 187
About Test Cases on page 164

Class IDs Test Case

➤ Quality Assurance module only.

If Test Expert detects Class IDs in a package, the Class IDs test case appears. It checks whether COM objects declared by a ClassID can be created (instantiated). This test might expose problems such as missing .DLLs that are required by the ClassID to create the object.

This test is run automatically by Test Expert and can be run only if the package being tested is currently installed.

To run the test case

1. Select the Class IDs test case in the left pane of Test Expert.
   All class IDs installed by the package are listed.
2. Mark the check boxes of the class IDs to test.
3. Click Execute.
   Test Expert examines the class ID descriptions, tries to create class objects, and marks each test item as either passed or failed.
   When the tests are finished, the dialog box closes and the statuses are displayed in the Status column.
4. Select the overall status of the test case in Status of Test Case.

To troubleshoot failures

To see the returned error, double-click an item and look in the Test Details field on the Test Item Details dialog box.

Failed items indicate COM failures, which can happen for a variety of reasons, such as missing self-registration information, incompatible or missing files, missing licence files (.LIC), and search paths that are not valid. Make sure that .DLLs or .EXEs for the Class ID exist, as well as any dependent .DLL files.

See also:

About Test Cases on page 164
Application Verification Tests on page 180
File Extensions Test Case

▶ Quality Assurance module only.

If Test Expert detects file extensions in a package, the File Extensions test case appears. This test creates a list of verbs associated with each extension in the package. It prompts you to select a file of the type being tested. Then it tries to execute each verb on the file. You must visually determine whether the verb performs as expected.

Example: You can associate the verb Open with a file extension. This means that if the end user right-clicks on a file of the specified type, the word Open appears in the right-click menu.

Requirements

- The package must be installed on the testing computer.
- A file of each type specified must be available on the testing computer.

To run the test case

1. Select the File Extensions test case in the left pane of Test Expert.
   All verbs associated with each extension are listed.
2. Mark the check boxes of the verbs to test.
3. Click Execute.
   The Test Case - File Extensions dialog box appears, where you execute one test at a time.
4. Click Execute in the Test Case dialog box.
   An Open file dialog box appears.
5. Navigate to and open a file of the type specified in Files of type.
   Test Expert opens the file in the application and applies the verb. (Example: The verb Print might open the file, print it, and close the application, depending on how it was coded.) Typically, the verb is either Open or <default>, both of which open the file in the associated application.
6. When the Pass/Fail dialog box appears:
   - Click Pass or Fail based on your visual verification of the test results.
   - In Comments, enter any pertinent information about the test case.
7. In the Test Case dialog box, continue to click Execute for each subsequent test case, entering You cannot add rules to predefined validation modules results for each.
   When the tests are finished, the dialog box closes and the statuses are displayed in the Status column.
8. Select the overall status of the test case in Status of Test Case.

To troubleshoot failures

If it is an .MSI, in Windows Installer Editor, double-click the problem extension on the File Associations page. On the File Association Details dialog box, click the Command Verb tab and verify that the command-line options and arguments are correct.
To verify that the application accepts the command-line option and argument, run it from the command line. If it fails from the command line, the command-line option and argument are incorrect, or the program is not coded to accept them.

See also:

- About Test Cases on page 164
- Application Verification Tests on page 180

**Help Files Test Case**

> Quality Assurance module only.

If Test Expert detects help files in a package, the Help Files test case appears. This test searches the package for .HLP or .CHM files and lists them in the right pane. It verifies that the files can be opened.

This test can be run only if the package being tested is currently installed.

**To run the test case**

1. Select the Help Files test case in the left pane of Test Expert.
   - All help files detected in the package are listed.
2. Mark the check boxes of the help files to test.
3. Click Execute.
   - The Test Case - Help Files dialog box appears, where you execute one test at a time.
4. Click Execute in the Test Case dialog box.
   - After a pause, the installed help file should open.
5. Close the help file.
6. When the Pass/Fail dialog box appears:
   - a. Click Pass or Fail based on your visual verification of the test results.
   - b. In Comments, enter any pertinent information about the test case.
7. In the Test Case dialog box, continue to click Execute for each subsequent test case, entering Pass/Fail results for each.
   - When the tests are finished, the dialog box closes and the statuses are displayed in the Status column.
8. Select the overall status of the test case in Status of Test Case.

**To troubleshoot failures**

Investigate what files, operating system, or other system requirements are necessary to open help files.

See also:

- About Test Cases on page 164
- Application Verification Tests on page 180
**ODBC Data Sources Test Case**

➤ *Quality Assurance module only.*

If Test Expert detects ODBC data sources in a package, the ODBC Data Sources test case appears. Use it to check whether an ODBC data source has been created correctly and whether the data source is accessible.

This test can be run only if the package being tested is currently installed.

**To run the test case**

1. Select the ODBC Data Sources test case in the left pane of Test Expert.

   All ODBC data sources installed by the package are listed.

2. Mark the check boxes of the ODBC data sources to test.

3. Click Execute.

   Test Expert examines the ODBCDataSource table and then examines the data sources that are registered with ODBC. For each data source, the test item is marked passed if the data source is installed and could be connected to.

   When the tests are finished, the dialog box closes and the statuses are displayed in the Status column.

4. Select the overall status of the test case in **Status of Test Case**.

**To troubleshoot failures**

To see the returned error, double-click an item and look in the **Test Details** field on the Test Item Details dialog box.

If an ODBC test item fails for an .MSI, open the package in Windows Installer Editor and check the ODBCDataSource table. Make sure the data source was actually installed.

See also:

- *About Test Cases* on page 164
- *Application Verification Tests* on page 180

**Prog IDs Test Case**

➤ *Quality Assurance module only.*

If Test Expert detects ProgIDs in a package, the Prog IDs test case appears. Use it to check whether COM objects declared by a ProgID can be created (instantiated). This test might expose problems such as missing .DLLs that are required by the ProgID to create the object.

This test is run automatically by Test Expert. It can be run only if the package being tested is currently installed.

**To run the test case**

1. Select the Prog IDs test case in the left pane of Test Expert.

   All ProgIDs installed by the package are listed.

2. Mark the check boxes of the ProgIDs to test.
3. Click Execute.

Test Expert examines the ProgID descriptions, tries to create objects based on those ProgIDs, and then marks each test as either passed or failed.

When the tests are finished, the dialog box closes and the statuses are displayed in the Status column.

4. Select the overall status of the test case in **Status of Test Case**.

**To troubleshoot failures**

To see the returned error, double-click an item and look in the **Test Details** field on the Test Item Details dialog box.

Items that fail indicate COM failures, which can happen for a variety of reasons, such as missing self-registration information, incompatible or missing files, missing licence files (.LIC), and search paths that are not valid. Make sure that .DLLs or .EXEs for the Class ID exist, as well as any dependent .DLL files.

See also:

- *About Test Cases* on page 164
- *Application Verification Tests* on page 180

**Search Locations Test Case**

*Quality Assurance module only.*

If Test Expert detects search locations in a package, the Search Locations test case appears. This test verifies that search locations defined within the package exist and are accessible. It does not verify the presence of the package’s source files.

Search locations represent locations the package will search for source files if the installation is run in maintenance mode. In Windows Installer Editor, you can view search locations for a package in Installation Expert > Search Locations page.

This test is run automatically by Test Expert. Because the accessibility of shared network locations often depends on user privileges, this test case should be run while users of limited privileges are logged on.

**To run the test case**

1. Select the Search Locations test case in the left pane of Test Expert.

All search locations installed by the package are listed.

2. Mark the check boxes of the search locations to test.

3. Click Execute.

Test Expert examines the search locations, tries to access each, and then marks each test as either passed or failed.

When the tests are finished, the dialog box closes and the statuses are displayed in the Status column.

4. Select the overall status of the test case in **Status of Test Case**.
To troubleshoot failures
To see the returned error, double-click an item and look in the Test Details field on the Test Item Details dialog box.

If a search location does not exist or is not accessible, either open the package and remove the search location, or work with your system administrator to make the location accessible to the lowest privilege level on which you expect the installation to run.

See also:
About Test Cases on page 164
Application Verification Tests on page 180

Services Test Case

Quality Assurance module only.

If Test Expert detects services in a package, the Services test case appears. This test verifies that a service is installed and correctly registered with the computer’s Services Manager. It does not verify whether the service runs correctly.

This test is run automatically by Test Expert. It can be run only if the package being tested is currently installed.

To run the test case
1. Select the Services test case in the left pane of Test Expert.
   All services installed by the package are listed.
2. Mark the check boxes of the services to test.
3. Click Execute.
   Test Expert examines the ServiceInstall table and then examines the services that are registered with the computer’s Services Manager. For each service in the ServiceInstall table, the test item is marked Passed if the service is registered on the computer; otherwise it is marked Failed.
   When the tests are finished, the dialog box closes and the statuses are displayed in the Status column.
4. Select the overall status of the test case in Status of Test Case.

To troubleshoot failures
Double-click the failed test case to see the returned error, which is listed in Test Details.

If a services test item fails for an .MSI, open the package in Windows Installer Editor and check the ServiceInstall table. Check to see what component the service is a part of, and if the component has a condition that might have caused the component to not be installed.

See also:
About Test Cases on page 164
Application Verification Tests on page 180
Shortcuts Test Case

- **Quality Assurance module only.**

If Test Expert detects shortcuts in a package, the Shortcuts test case appears. This test verifies that shortcuts in the installation are attached to a valid target file and that the target opens when the shortcut is opened.

This test can be run only if the package being tested is currently installed.

**To run the test case**

1. Select the Shortcuts test case in the left pane of Test Expert.
   
   All shortcuts installed by the package are listed.
2. Mark the check boxes of the shortcuts to test.
3. Click Execute.
   
   The Test Case - Shortcuts dialog box appears, where you execute one test at a time.
4. Click the Execute button in the Test Case dialog box.
   
   After a pause, the shortcut referenced by the test should open.
5. Close the program opened by the shortcut.
6. When the Pass/Fail dialog box appears:
   
   a. Click Pass or Fail based on your visual verification of the test results.
   b. In **Comments**, enter any pertinent information about the test case.
7. In the Test Case dialog box, continue to click Execute for each subsequent test case, entering Pass/Fail results for each.
   
   When the tests are finished, the dialog box closes and the statuses are displayed in the Status column.
8. Select the overall status of the test case in **Status of Test Case**.

**To troubleshoot failures**

If a Shortcut test item fails for an .MSI, open the package in Windows Installer Editor and check Installation Expert > Shortcuts page. View the details for the shortcut that failed, particularly the command-line options that are displayed in **Arguments**.

See also:

*About Test Cases* on page 164

*Application Verification Tests* on page 180

Application Execution Tests

- **Quality Assurance module only.**

In Test Expert, application execution tests are run by running the installed application and exercising all of its functions.

All tests in the Application Execution Tests group are run simultaneously while you run the application. You cannot test only one test case at a time, and it is unlikely that you
would want to do so, because running each test requires a complete exercise of the application's functions.

During application execution tests, an Application Monitor window appears. A series of graphs represent the percentage of items that have been accessed so far during testing. For details of each graph, see the individual test case descriptions.

See [How to Run Application Execution Tests](#) on page 189.

**Note**

A security setting in Windows Vista or later operating system prevents application execution tests from working. When you try to run a test, a prompt appears and provides the option to disable the security restriction. If you choose to disable the security restriction, the following registry setting is set:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Windows>LoadAppInit_DLLs=1
```

The initial default for this setting is 0. If you disable this restriction, your computer's vulnerability to malicious attack is increased. However, if you run Test Expert in a testing environment, the increased vulnerability might not be a critical issue.

**Requirements**

- Application execution tests require the application to be installed. Run the installation tests to install the application.

- It is preferable to install the application either on a clean machine or on a computer that contains the default software common to all computers on your network.

- The Application Monitor window must remain open while you are running the application, and must be closed when you finish. Closing the window stops the monitoring of the testing computer.

**Tips**

- If an error occurs when a file is accessed, the status is set to Fail and the error from the operating system is displayed in the Error column. This lets you track such things as missing files and user rights lacking the necessary permissions.

- Application execution tests are primarily informational. Example: The Extra Files test case might reveal that a printing-related system file is accessed when you print. This is not necessarily a failure.

- You might notice that no matter how thoroughly you exercise the application, some files are not accessed. This could be because some files are not meant to be accessed as part of normal functioning. Example: If data files are located in a Samples directory, they are not opened as part of the normal functioning of the application.

- Some errors might appear as a result of the application design. Example: Suppose that, to maintain backward compatibility, an application is written to check an old registry location, then check a new location. During testing, the first registry location might appear as an error. This merely means that the old version of the application was never installed on the testing computer.

See also:

- [Extra Files Test Case](#) on page 190
- [Extra Registry Entries Test Case](#) on page 191
How to Run Application Execution Tests

This procedure applies to all application execution tests.

---

**Note**
A security setting in Windows Vista or later operating system prevents application execution tests from working. When you try to run a test, a prompt appears and provides the option to disable the security restriction. If you choose to disable the security restriction, the following registry setting is set:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Windows>LoadAppInit_DLLs=1
```

The initial default for this setting is 0. If you disable this restriction, your computer’s vulnerability to malicious attack is increased. However, if you run Test Expert in a testing environment, the increased vulnerability might not be a critical issue.

---

**To run application execution tests**

1. Make sure the application is installed by running the Verify Installation test case in the Installation Tests group.
   
   See [Verify Installation Test Case](#) on page 177.

2. Close all applications other than Wise Package Studio and Test Expert, including all background applications and services that might access files or registry entries on the testing computer. Files or registry entries accessed by other applications can interfere with the test results.

3. Select any of the test cases in the Application Execution Tests group in the left pane of Test Expert.

4. Click one of the following buttons:
   
   - **Run**
     
     Run the application normally.

   - **Run As**
     
     Run the application as a user with different privileges. A dialog box opens, in which you specify the user name, domain, and password of a user whose privileges the application will be run under. The dialog box differs depending on the operating system. Click OK.

     The Run Application Execution Tests dialog box appears and lists all .EXEs in the package that have shortcuts.

5. In the Run Application Execution Tests dialog box, select one of the following and click OK.
   
   - **Select a file to run from shortcuts installed by this package**
     
     Select an .EXE from the list of shortcuts to run its target file.

   - **Specify a file to run**
     
     If the file you want to run doesn’t have an installed shortcut, select this option and browse to a file. This file should run the application you are testing.
Typically, you will select an .EXE, but you can also select any other file type. (Example: If the package installs an HTML help system, you might open an .HTM or .ASP file.) To open non-.EXE files, change Files of type to All Files.

6. When the application starts, exercise all functions of the application. This might include selecting every menu option, creating, editing, and saving every type of file, changing options on all dialog boxes, and so on. The more complex the application, the longer this can take.

The Application Monitor window appears. If you minimize this window, click the Monitor button in the Test Expert toolbar to maximize it. Do not close the Application Monitor window until you finish testing.

7. When you finish testing, exit the application and close the Application Monitor window.

**Note**
It is important to close the Application Monitor window after closing the application, because monitoring of your system continues until the window is closed.

8. Review the test items for each application execution test case. For information on interpreting the results, see the individual test case descriptions.

9. When the test is finished, select the status for each test item based on your visual verification of the test results.

10. Select the overall status of the test case in **Status of Test Case**.

See also:

*Setting Test_statuses and Details* on page 166
*Extra Files Test Case* on page 190
*Extra Registry Entries Test Case* on page 191
*File Coverage Test Case* on page 191
*Isolated Files Test Case* on page 192
*Registry Coverage Test Case* on page 193

**Extra Files Test Case**

➢ **Quality Assurance module only.**

The Extra Files test case in Test Expert is primarily informational. It monitors the testing computer as you exercise the features of the application. It then identifies files that are accessed by the application but that are not installed by the package. Because all application execution tests are run at the same time, the test items for every test case in the Application Execution Tests group are updated simultaneously.

Before you run this test, close all applications other than Wise Package Studio and Test Expert, including all background applications and services that might access files or registry entries on the testing computer.

Initially, when you select this test case, you see no test items. As you run the application, non-installed files are added to the list with the status of Pending. Installed files that are accessed are added to the File Coverage test with a Passed status. You must change the status for each item manually.

During this test, you check whether any unexpected files are accessed by the application. You also see whether files were unable to be accessed by the application,
and the system error that resulted. In general it is not an error if an application accesses shared files. It could become a problem if the application depends on the file but makes no attempt to install it if it is missing.

The Application Monitor window shows the extra files accessed as a percentage of the number of files that are installed. Example: If the package installs 100 files, and during execution 10 non-installed files are accessed, the Extra Files graph shows 10%.

See also:

*How to Run Application Execution Tests* on page 189  
*About Test Cases* on page 164

**Extra Registry Entries Test Case**

➤ *Quality Assurance module only.*

The Extra Registry Entries test case in Test Expert is primarily informational. It monitors the testing computer as you exercise the features of the application. It then identifies registry entries that are accessed by the application but that are not installed by the package. Because all application execution tests are run at the same time, the test items for every test case in the Application Execution Tests group are updated simultaneously.

Before you run this test, close all applications other than Wise Package Studio and Test Expert, including all background applications and services that might access files or registry entries on the testing computer.

Initially, when you select this test case, you see no test items. As you run the application, non-installed registry entries are added to the list with the status of Pending. Installed registry entries that are accessed are added to the Registry Coverage test with a Passed status. You must change the status for each item manually.

During this test, you check whether any unexpected registry entries are accessed by the application. You also see whether registry entries were unable to be accessed by the application, and the system error that resulted. In general it is not an error if an application accesses shared registry entries. It could become a problem if the application depends on the registry entry but makes no attempt to install it if it is missing.

The Application Monitor window shows the extra registry entries accessed as a percentage of the number of registry entries that are installed. Example: If the package installs 100 registry entries, and during execution 10 non-installed registry entries are accessed, the Extra Registry Entries graph shows 10%.

See also:

*How to Run Application Execution Tests* on page 189  
*About Test Cases* on page 164

**File Coverage Test Case**

➤ *Quality Assurance module only.*

The File Coverage test case in Test Expert monitors the testing computer as you exercise the features of the application. It then records what installed files are accessed during application execution. Because all application execution tests are run at the same
time, the test items for every test case in the Application Execution Tests group are updated simultaneously.

Before you run this test, close all applications other than Wise Package Studio and Test Expert, including all background applications and services that might access files or registry entries on the testing computer.

Initially, when you select this test case, you see a list of all files installed by the application. Each has the status of Pending. In the Test Expert window, accessed files are given the status of Passed.

Because a file was not accessed does not necessarily mean that the test failed. Example: The package might install sample files in a Samples directory, which is not accessed unless the end user opens them. Or perhaps a file is only accessed under very specific circumstances, and you have not duplicated those circumstances with your testing. Use your own judgement in setting the statuses of files that are not accessed during installation.

The Application Monitor window shows the percentage of installed files accessed. Example: If the package installs 100 files, and during execution 90 of those files are accessed, the File Coverage graph shows 90%.

See also:
How to Run Application Execution Tests on page 189
About Test Cases on page 164

**Isolated Files Test Case**

*Quality Assurance module only.*

If Test Expert detects file extensions in a package, the Isolated Files test case appears. It monitors the testing computer as you exercise the features of the application. It then records what isolated files are accessed during application execution. Because all application execution tests are run at the same time, the test items for every test case in the Application Execution Tests group are updated simultaneously.

Before you run this test, close all applications other than Wise Package Studio and Test Expert, including all background applications and services that might access files or registry entries on the testing computer.

Initially, when you select this test case, you see a list of all isolated files installed by the application. Each has the status of Pending. As you run the application, isolated files that are accessed are given the status of Passed.

During this test, you check how the application accesses the isolated file. When a .DLL is isolated with an .EXE, the .DLL is stored in the application directory instead of the system directory. The desired behavior of isolated files is that the isolated copy should be accessed, and the copy in the system directory should not be accessed.

The Application Monitor window does not display a graph for isolated files.

See also:
How to Run Application Execution Tests on page 189
About Test Cases on page 164
Registry Coverage Test Case

➢ Quality Assurance module only.

The Registry Coverage test case in Test Expert monitors the testing computer as you exercise the features of the application. It then records what installed registry entries are accessed during application execution. Because all application execution tests are run at the same time, the test items for every test case in the Application Execution Tests group are updated simultaneously.

Before you run this test, close all applications other than Wise Package Studio and Test Expert, including all background applications and services that might access files or registry entries on the testing computer.

Initially, when you select this test case, you see a list of all registry entries installed by the application. Each has the status of Pending. As you run the application, accessed registry entries are given the status of Passed.

The Application Monitor window shows the percentage of installed registry entries that are accessed. Example: If the package installs 100 registry entries, and during execution 90 of those registry entries are accessed, the Registry Coverage graph shows 90%.

See also:

How to Run Application Execution Tests on page 189
About Test Cases on page 164

Uninstall Tests

➢ Quality Assurance module only.

In Test Expert, run uninstall tests to determine how the uninstall sequence in the package is performing. All tests in the Uninstall Tests group are run simultaneously when you uninstall the application.

Note
Uninstall tests do not appear if a group is open.

Uninstall tests use the pre-installation snapshot created by Machine Capture. When the application is uninstalled, you use Machine Capture to create a post-uninstall snapshot. The differences between the two snapshots provide the following information:

- **Created items**
  Items that were not present before installation, and were not installed, but were present after installation. This includes all files and registry items created by the application when it was opened, as well as any files you created while testing.

- **Destroyed items**
  Items that were present before installation and were not present after installation. In general, installations should not remove anything that they did not install because of the danger of breaking other applications.

- **Residual items**
  Items that were installed but were not removed. This is the most common uninstall problem and is not usually considered to be serious.
You can configure Machine Capture settings to limit the areas of the computer that are captured in the snapshot.

See *Machine Capture Settings* on page 169.

Installation tests and uninstall tests do not appear if you opened a cached copy of an .MSI. Every time a Windows Installer installation is run, a randomly-named copy of the .MSI is cached in the system directory. The cached copy contains the installation information and logic, but it does not contain the installation files.

See also:
- *Created Files Test Case* on page 195
- *Created Registry Entries Test Case* on page 196
- *Destroyed Files Test Case* on page 197
- *Destroyed Registry Entries Test Case* on page 198
- *Residual Files Test Case* on page 198
- *Residual Registry Entries Test Case* on page 199
- *About Test Cases* on page 164

### How to Run Uninstall Tests

> **Quality Assurance module only.**

This procedure applies to all uninstall tests.

**Note**

Uninstall tests do not appear if a group is open.

**Before running uninstall tests**

1. Click the Install or Install As button that appears on the Test Expert toolbar.
2. Create a pre-installation snapshot with Machine Capture.
3. Install the application.
   
   For details on the preceding steps, see *How to Run Installation Tests* on page 174.
4. Run application execution tests, running the installed application and exercising all of its functions.
   
   See *How to Run Application Execution Tests* on page 189.
5. If the application was installed into a virtual software layer, verify that the layer is activated.

**To run uninstall tests**

1. Close all applications other than Wise Package Studio and Test Expert, including all background applications and services that might access files and registry entries on the testing computer.

   Files or registry entries created by other applications can interfere with the test results.
2. After you finish the application execution tests, select any uninstall test in the left pane and click Uninstall.
Note
Rerunning the uninstall replaces previous uninstall test results.

An uninstall is performed, and then the Welcome dialog box of Machine Capture appears.

3. On the Welcome dialog box, click Next.

The Capturing Machine State dialog box appears and the scan begins, which takes a few moments. This creates a post-uninstall snapshot.

4. When the capture is complete, click Next.

The Comparing Machine States dialog box appears while the pre-installation and post-uninstall snapshots are compared. When the comparison is finished, the dialog box closes.

All uninstall tests are populated with uninstall test results.

5. Select the status for each test item based on your visual verification of the test results.

6. Select the overall status of the test case in Status of Test Case.

See also:

Setting Test Statuses and Details on page 166
Created Files Test Case on page 195
Created Registry Entries Test Case on page 196
Destroyed Files Test Case on page 197
Destroyed Registry Entries Test Case on page 198
Residual Files Test Case on page 198
Residual Registry Entries Test Case on page 199

Created Files Test Case

➤ Quality Assurance module only.

The Created Files test case in Test Expert, which is primarily informational, shows files that were created after installation. After uninstall tests are run, the test case list is populated with created files and all statuses are set to Pending. Review the list and select the status for each test item.

Note
Uninstall tests do not appear if a group is open.

Because all uninstall tests are run at the same time, the test items for every test case in the Uninstall Tests group are updated simultaneously.

This test compares the pre-installation snapshot to the post-uninstall snapshot and reports the differences between the two snapshots. Files accessed by other applications can interfere with the test results. Before you run this test, close all applications other than Wise Package Studio and Test Expert, including all background applications and services that might access files or registry entries on the testing computer.

It is common for applications to create files, so files listed here do not necessarily represent errors. If files were created that should not have been created, check custom actions in the package. To do this, open the package in Windows Installer Editor and
examine custom actions that appear in MSI Script. Sometimes custom actions call processes during installation that result in created files.

If the package creates files that should be removed on uninstall, you can set the package to remove them, as long as the files have fixed names. To do this, open the package in Windows Installer Editor. Add the file name to the RemoveFile table on the Setup Editor > Tables tab.

See also:
* How to Run Uninstall Tests on page 194
* About Test Cases on page 164

**Created Registry Entries Test Case**

* Quality Assurance module only.

The Created Registry Entries test in Test Expert, which is primarily informational, shows registry entries that were created after installation. After uninstall tests are run, the test case list is populated with created registry entries and all statuses are set to Pending. Review the list and select the status for each test item.

**Note**

Uninstall tests do not appear if a group is open.

Because all uninstall tests are run at the same time, the test items for every test case in the Uninstall Tests group are updated simultaneously.

This test compares the pre-installation snapshot to the post-uninstall snapshot and reports the differences between the two snapshots. Registry entries accessed by other applications can interfere with the test results. Before you run this test, close all applications other than Wise Package Studio and Test Expert, including all background applications and services that might access files or registry entries on the testing computer.

It is common for applications to create registry entries, so those listed here do not necessarily represent errors. If registry entries were created that should not have been created, check custom actions in the package. To do this, open the package in Windows Installer Editor and examine custom actions that appear in MSI Script. Sometimes custom actions call processes during installation that result in created registry entries.

If the package creates registry values that should be removed on uninstall, you can set the package to remove them, as long as the registry values have fixed names. To do this, open the package in Windows Installer Editor. Add the registry value name to the RemoveRegistry table on the Setup Editor > Tables tab.

See also:
* How to Run Uninstall Tests on page 194
* About Test Cases on page 164
Destroyed Files Test Case

➤ Quality Assurance module only.

The Destroyed Files test case in Test Expert shows files that existed on the computer before installation, but were missing after uninstall.

**Note**

Uninstall tests do not appear if a group is open.

After uninstall tests are run, the test case list is populated with destroyed files and all statuses are set to Pending. Review the list and select the status for each test item.

Because all uninstall tests are run at the same time, the test items for every test case in the Uninstall Tests group are updated simultaneously.

This test compares the pre-installation snapshot to the post-uninstall snapshot and reports the differences between the two snapshots. Files accessed by other applications can interfere with the test results. Before you run this test, close all applications other than Wise Package Studio and Test Expert, including all background applications and services that might access files or registry entries on the testing computer.

Normally, an installation should not uninstall files that it did not install. If it does, this is a serious error because removing files that another application installed could potentially break the other application.

This error can be caused when two separate packages install the same file to the same place, but the file has a different component ID in each package. Example: Suppose Package1 installs sample.dll into the System32 directory, then Package2 installs a newer version of sample.dll over the original. When Package2 is uninstalled, it removes sample.dll, breaking Package1.

**How to Fix Errors**

- The best method is to isolate the file using Application Isolation in Wise Package Studio. This tool isolates shared .DLLs in the application directory of the application that uses it, ensuring that each application uses its own version of the file. See Application Isolation on page 89.

- If you have access to both packages that install the file, and the file is really the same in each, try aligning the component IDs of the components that contain the file. Open the package in Windows Installer Editor and align component IDs on the Setup Editor > Components tab.

- Set the .MSI to leave the file installed during uninstall. To do this, open the package in Windows Installer Editor. Display the component’s details on the Setup Editor > Components tab and mark Leave installed on uninstall. This has the disadvantage of bloating the computer’s contents, because the file will not be uninstalled by the other application either.

See also:

* How to Run Uninstall Tests on page 194
* About Test Cases on page 164
Destroyed Registry Entries Test Case

))-> Quality Assurance module only.

The Destroyed Registry Entries test case in Test Expert shows registry entries that existed on the computer before installation, but were missing after uninstall. After uninstall tests are run, the test case list is populated with destroyed registry entries and all statuses are set to Pending. Review the list and select the status for each test item.

**Note**

Uninstall tests do not appear if a group is open.

Because all uninstall tests are run at the same time, the test items for every test case in the Uninstall Tests group are updated simultaneously.

This test compares the pre-installation snapshot to the post-uninstall snapshot and reports the differences between the two snapshots. Registry entries accessed by other applications can interfere with the test results. Before you run this test, close all applications other than Wise Package Studio and Test Expert, including all background applications and services that might access files or registry entries on the testing computer.

Normally, an installation should not uninstall registry entries that it did not install. This is a serious error because removing registry entries that another application installed could potentially break the other application.

This error can be caused when two separate packages install the same registry entry to the same place, but the registry entry has a different component ID in each package. Example: Suppose Package1 installs a registry value, then Package2 installs the same value to the same place. When Package2 is uninstalled, it removes the registry key, breaking Package1.

**How to fix errors**

- If you have access to both packages that install the registry value, try aligning the component IDs of the components that contain the registry value. Open the packages in Windows Installer Editor and align component IDs on the Setup Editor > Components tab.

- Set the .MSI to leave the registry value installed during uninstall. To do this, open the package in Windows Installer Editor. Display the component’s details on the Setup Editor > Components tab and mark Leave installed on uninstall. This has the disadvantage of bloating the computer’s contents, because the registry value will not be uninstalled by the other application either.

See also:

*How to Run Uninstall Tests* on page 194

*About Test Cases* on page 164

Residual Files Test Case

))-> Quality Assurance module only.

The Residual Files test case in Test Expert shows files that did not get uninstalled properly. After uninstall tests are run, the test case list is populated with residual files.
and all statuses are set to Pending. Review the list and select the status for each test item.

**Note**
Uninstall tests do not appear if a group is open.

Because all uninstall tests are run at the same time, the test items for every test case in the Uninstall Tests group are updated simultaneously.

This test compares the pre-installation snapshot to the post-uninstall snapshot and reports the differences between the two snapshots. Files accessed by other applications can interfere with the test results. Before you run this test, close all applications other than Wise Package Studio and Test Expert, including all background applications and services that might access files or registry entries on the testing computer.

Normally, an installation should delete all files and registry entries that it created. When it doesn’t, it generally is a minor error.

This error can be caused when the component that installs the file is set to never uninstall. To check this, open the package in Windows Installer Editor. On the Setup Editor > Components tab, display the details for the component that contains the residual file. If the **Leave installed on uninstall** check box is marked, the file is set to never uninstall. Clear the check box.

See also:
* How to Run Uninstall Tests on page 194
* About Test Cases on page 164

**Residual Registry Entries Test Case**

*Quality Assurance module only.*

The Residual Registry Entries test case in Test Expert shows registry entries that did not get uninstalled properly. After uninstall tests are run, the test case list is populated with residual registry entries and all statuses are set to Pending. Review the list and select the status for each test item.

**Note**
Uninstall tests do not appear if a group is open.

Because all uninstall tests are run at the same time, the test items for every test case in the Uninstall Tests group are updated simultaneously.

This test compares the pre-installation snapshot to the post-uninstall snapshot and reports the differences between the two snapshots. Registry entries accessed by other applications can interfere with the test results. Before you run this test, close all applications other than Wise Package Studio and Test Expert, including all background applications and services that might access files or registry entries on the testing computer.

Normally, an installation should delete all files and registry entries that it created. When it doesn’t, it generally is a minor error.

This error can be caused when the component that installs the registry value is set to never uninstall. To check this, open the package in Windows Installer Editor. On the Setup Editor > Components tab, display the details for the component that contains the
residual registry value. If **Leave installed on uninstall** is marked, the registry value is set to never uninstall. Clear the check box.

See also:

*How to Run Uninstall Tests* on page 194  
*About Test Cases* on page 164
Chapter 8
Capturing Applications

This chapter includes the following topics:

- About Capturing Applications on page 201
- SetupCapture Configuration on page 202
- SetupCapture on page 216
- Using SetupCapture With Virtual Capture on page 237
- Using SetupCapture to Capture First Use Settings on page 240
- SOE Snapshot on page 242
- Capturing With Wise Web Capture on page 245
- Files and Registry Entries Ignored During Captures on page 247

About Capturing Applications

Part of the process of repackaging is converting an installation to a format that you can manipulate to meet your corporate standards. SetupCapture® records all the changes performed by an installation and saves that information to a new Windows Installer, WiseScript, or virtual software package. Then you can edit the package or import it into the Software Manager database so you can compare it to other applications and, if needed, resolve potential conflicts. Alternatively, you can use SetupCapture to capture the first use changes that an application makes to a computer.

Wise Web Capture, which you run from a browser, lets you capture installations on a clean machine without installing any additional software. It also lets you capture on a computer that is running a non-supported system.

SOE Snapshot is used exclusively in conjunction with Software Manager and ConflictManager. SOE Snapshot lets you capture a computer's standard operating environment (SOE). SOE Snapshot stores this snapshot in the form of an .SOE file, which you can import into the Software Manager database to represent a baseline computer in your organization. SOE Snapshot is not available in Standard Edition.

SetupCapture Configuration lets you edit and create configuration files that control how SetupCapture and SOE Snapshot work. You select the configuration file when you run SetupCapture or SOE Snapshot. SetupCapture Configuration does not apply when the captured application is saved as a virtual software package.

See:

SetupCapture Configuration
SetupCapture
Using SetupCapture With Virtual Capture
Using SetupCapture to Capture First Use Settings
SOE Snapshot
Capturing With Wise Web Capture
SetupCapture Configuration

SetupCapture Configuration lets you edit and create configuration files that control how SetupCapture and SOE Snapshot work. You select the configuration file when you run SetupCapture or SOE Snapshot. SetupCapture Configuration does not apply when the captured application is saved as virtual software package or when using Wise Web Capture.

You also can edit a configuration file from within SetupCapture or SOE Snapshot by clicking Settings.

Configuration files are stored as .INI files. Two default configuration files are provided:

- WisePSSC.ini, located in the Windows or Winnt directory on the local computer.
- repackage.ini, located in the share point directory. This lets you share central, standardized settings.

You can edit the default configuration files or create additional configuration files that are optimized for specific purposes. Although you can use the same configuration file for SetupCapture and SOE Snapshot, you might want to create a different configuration file for each tool. Example: SOE Snapshot uses only the exclusions settings, so you might want to create a separate configuration file with different exclusions settings for SOE Snapshot.

Machine Capture in Test Expert uses the local configuration file when it creates pre-installation and post-uninstall snapshots. Changes you make on the Machine Capture Settings dialog box affect the local configuration file.

SetupCapture Configuration lets you specify:

- The location of the configuration file.
  See Selecting the Configuration File on page 205.
- General settings that determine what kind of information is captured and how that information is interpreted in the repackaged installation.
  These settings apply to SetupCapture only.
  See Setting General Settings on page 205.
- The method to use for capturing changes during an installation: Virtual Capture™, snapshot, or SmartMonitor™. SOE Snapshot always uses the snapshot method.
  See Setting General Settings on page 205 and Selecting the Capture Methodology on page 224.
- The directories to scan or monitor for changes during SetupCapture.
  These options apply to SetupCapture only; SOE Snapshot always scans the entire computer.
  See Setting Directories to Watch on page 208.
- The files or directories in which changes should be ignored.
  See Setting File and Folder Exclusions on page 211.
- The registry values or keys to ignore.
  See Setting Registry Exclusions on page 214.
- .INI files or parts of .INI files to ignore.
  See Setting INI File Exclusions on page 215.
Configuring Settings in SetupCapture Configuration

Use SetupCapture Configuration to edit and create configuration files that control how SetupCapture and SOE Snapshot work. SOE Snapshot uses only the exclusions settings.

1. Do one of the following:
   a. On the Projects tab, click the Run link to the right of the task or tool associated with SetupCapture Configuration.
   b. On the Tools tab, double-click SetupCapture Configuration.
      The Welcome page appears.

2. To select a configuration file, click Change. All settings you change are stored in this file.
   See Selecting the Configuration File on page 205.
   If the Change button is unavailable, you might not have permission to change settings.
   See Setting SetupCapture Configuration Security on page 44.

3. Click Next on the Welcome page.
   The General Settings page appears.

4. On the General Settings page, edit settings that determine how captures are performed, then click Next. These settings apply to SetupCapture only.
   See Setting General Settings on page 205.
   The Directories to Watch page appears. This page applies to SetupCapture only.

5. If you are using the snapshot capture method, specify the directories to scan for changes during SetupCapture and click Next.
   If you automatically build an exclusion list later in SetupCapture Configuration, the directories you specify are scanned during the exclusion list building process. The recommended setting is the root drive (C:\) with Include Sub-directories turned on. Changing the default is not recommended.
   See Setting Directories to Watch on page 208.
   SmartMonitor monitors all local directories, regardless of what is specified in Directories to Watch.
   The Build Exclusion List page appears.

6. To build an exclusion list, mark Automatically build exclusion list (Recommended), click Next, and follow the prompts. Otherwise mark Do not automatically build exclusion list and click Next.
   See Building an Exclusion List Automatically on page 210.
   If you choose to build an exclusion list, additional dialog boxes appear:
   a. Begin Exclusion Capture. Click Next to begin scanning the registry and the directories specified in Directories to Watch.
Run Applications / Reboot. Run any applications that might be executed by an installation (examples: Internet Explorer, Notepad) and then restart the computer to ensure a complete exclusion list.

Leave the Run Applications / Reboot page open as you restart. Do not close it or click Cancel. After restart, SetupCapture Configuration reopens to the Run Applications / Reboot page. Click Next.

End Exclusions Capture. Click Next to rescan the computer.

After the exclusion list is built, the File and Folder Exclusions page appears. It displays the files and folders in the current configuration file’s exclusion list. Items with a question mark icon were added by the exclusion list building process. Items without a question mark icon were already in the exclusion list.

7. Specify files and folders to be ignored by SetupCapture or SOE Snapshot and click Next.

See Setting File and Folder Exclusions on page 211.

The Registry Exclusions page appears. It displays the registry entries in the current configuration file’s exclusion list. Items with a question mark icon were added by the exclusion list building process. Items without a question mark icon were already in the exclusion list.

8. Specify registry keys and values to exclude from a SetupCapture or SOE Snapshot and click Next.

See Setting Registry Exclusions on page 214.

The INI Files Exclusions page appears. It displays the .INI file items in the current configuration file’s exclusion list. Items with a question mark icon were added by the exclusion list building process. Items without a question mark icon were already in the exclusion list.

9. Specify .INI files or parts of .INI files to be ignored by a SetupCapture or SOE Snapshot and click Next.

See Setting INI File Exclusions on page 215.

10. In the Professional Edition, the Virtual OS Files page appears. It reminds you that, if you plan to use Virtual Capture, you must create a Virtual OS on a clean machine before you can run SetupCapture.

11. If the Virtual OS Files page appears, click Next.

The Finish page appears.

12. Click Finish, and the settings you specified in SetupCapture Configuration are saved to the configuration file you specified on the first page of SetupCapture Configuration.

To use this configuration file, click Change on the Welcome page of SetupCapture or SOE Snapshot.

See Selecting the Configuration File.

If you added one or more file exclusions, you are prompted to check existing packages for those files and remove them. If you click Yes, Software Manager opens and the Remove Excluded Files wizard runs.

See Removing Excluded Files From Packages in the Software Manager Help.
Selecting the Configuration File

Configuration files control how SetupCapture and SOE Snapshot work.

You select the configuration file from:

- SetupCapture or SOE Snapshot. The specified configuration file controls how SetupCapture or SOE Snapshot works during the current session.

- SetupCapture Configuration. The specified configuration file is edited or created.

Configuration files are stored as .INI files. They can be located on your local computer, in the share point directory, or in any other directory.

**Note**

Only share configuration files between computers that have identical software and hardware configurations, because exclusion lists and directories to watch are computer-dependent.

To select a configuration file

1. On the Welcome page of SetupCapture, SOE Snapshot, or SetupCapture Configuration, click Change. If the Change button is unavailable, you might not have permission to change configuration settings.

   See *Setting SetupCapture Configuration Security* on page 44.

   The Configuration File dialog box appears.

2. Select the location of the configuration file to use or edit.

   - **Use configuration file on share point**
     Edit or use the configuration file in the share point directory. The file is named repackage.ini. By default, it contains a recommended basic exclusion list. This lets you share central, standardized configuration settings with coworkers. This option might be unavailable.

     See *Setting SetupCapture Configuration Security* on page 44.

   - **Use configuration file from the local PC**
     Edit or use the configuration file on the local computer. The file is named WisePSSC.ini, located in the Windows or Winnt directory. By default, it contains a recommended basic exclusion list. Use the local configuration file if you don’t need to share configuration settings with coworkers.

   - **Use configuration file located in directory below**
     Create, edit, or use a configuration file in any directory accessible by your computer. Specify a new or existing file name. This option offers the most flexibility for storing and exchanging configuration files by letting you maintain multiple configuration files that are optimized for specific purposes. Example: You can create separate configuration files for SOE Snapshot and SetupCapture.

3. Click OK to close the Configuration File dialog box and return to the Welcome page of SetupCapture, SOE Snapshot, or SetupCapture Configuration.

Setting General Settings

Before you run SetupCapture, edit general settings that determine what kind of information is captured and how that information is interpreted in the repackaged installation that results from the SetupCapture.
Capturing Applications

The General Settings dialog box appears:

- During SetupCapture Configuration.
- When you run SetupCapture and click Settings on the Welcome page. If the Settings button is unavailable, you might not have permission to change settings.
  
  See Setting SetupCapture Configuration Security on page 44.

The general settings do not apply to SOE Snapshot.

General Settings

- **Include files deleted during capture**
  Normally, SetupCapture doesn’t record the deletion of files by an installation, which helps protect the destination computer from damage. Mark this to have SetupCapture record the deletion of files and add the necessary deletion code in the repackaged installation.

- **Include registry keys deleted during capture**
  Normally, SetupCapture doesn’t record the deletion of registry keys by an installation. Mark this to have SetupCapture record the deletion of registry keys and add the necessary deletion code in the repackaged installation.

- **Capture changes in hardware registry entries**
  Normally, SetupCapture doesn’t record changes to hardware registry entries, which are typically stored in HKEY_LOCAL_MACHINE\SYSTEM. Installing the same hardware information on destination computers that have different hardware could damage the system. Mark this only if all destination computers of the repackaged installation will have identical hardware configuration.

- **Allow root to be watched during capture**
  Normally, even if you select the root as a directory to watch, files on the root are ignored. Only its subdirectories are watched. Typically, you don’t want to edit the root’s system files because doing so can result in a damaged or inoperable operating system. To capture changes to files on the root, mark this check box. By itself, marking this check box does nothing—you must also select the root drive (example: C:) on the Directories to Watch dialog box.

- **Enable ordering of self-registration**
  (Windows Installer packages only.)
  This applies to the SmartMonitor capture method only. Some installations have files that must be self-registered in a specific order. Mark this to have SetupCapture detect the order in which these files were self-registered. It then replicates the order in the repackaged installation.

- **Enable checking of files against Wise Software Repository**
  (Windows Installer Packages only. Not available in Standard Edition.) This is marked by default. When SetupCapture adds a file that is used by a package in the Wise Software Repository, the Files in Repository dialog box appears and prompts you to add a version of the file that is in the repository.

  To disable the repository check, clear this check box. You can check the files in the repackaged installation against the repository after the capture is finished, by selecting Tools > Check Repository Files in Windows Installer Editor.

  See Adding Files From the Wise Software Repository in the Windows Installer Editor Help.
- **Capture non-Microsoft ODBC information**
  In most cases, you can leave this check box cleared. If this check box is cleared, ODBC information is added to the ODBC driver table (in Windows Installer packages) or added as an Install ODBC Driver action (in WiseScript packages).

  If you mark this check box, ODBC information is added as registry entries instead. Mark this check box only if:
  - You know the installation you’re capturing contains ODBC drivers, and
  - those drivers were not installed using Microsoft’s recommended technique.
    (Microsoft recommends that developers register their ODBC drivers using ODBCInst.dll.)

- **Create installation sequence report**
  (Not available in Standard Edition.)

  Mark this to generate a log file that shows the changes that the installation made in the order in which they occurred. This check box is unavailable if you are using snapshot comparisons only, because the snapshot method does not capture the order of installation, only the changes.

  This option is unrelated to the View Report button on the SetupCapture Inclusions/Exclusions dialog boxes, which generates a temporary report regardless of what you select here.

- **Enhanced File and Registry Key Association**
  Normally, SetupCapture associates a file and registry keys by reading the file’s self-registration information from the registry and from the file’s type library. When a file’s self-registration information and type library do not include all of the registry keys that the file updates during registration, SetupCapture may place those registry keys into different components than the file. (This mis-association can cause self-repair problems if one of the components is uninstalled.)

  When you mark this check box, SetupCapture performs an additional step (using the WiseComCapture.exe utility) to find all registry keys that are updated when a file is actually registered. With this additional information, SetupCapture moves registry keys updated by a file into the same component as the file, as necessary. If the additional step finds registry keys that were not found by the primary capture, those registry keys are not added to the repackaged installation. Using this feature speeds the installation that is created with SetupCapture.

  This feature registers every self-registering .DLL found in the capture. If you have a self-registering .DLL that you know is not set to be registered during the installation, then you might not want to use this feature because it could cause the repackaged installation to not function properly.

- **Advertising Setting (Windows Installer)**
  (Windows Installer packages only.)

  Windows Installer considers some kinds of registry entries, such as file extension definitions, to be advertising information.

  See Platform Support of Advertisement in the Windows Installer SDK Help.

  Specify how to handle advertising information:
  - **Retain registry information as-is**
    Try to create a package that does not support advertising. SetupCapture will not convert registry entries to advertising information, but instead leaves them as registry entries.
- **Convert registry entries into advertising info**
  Try to create a package that supports advertising. SetupCapture will convert registry entries to advertising information. It converts only those registry entries that are considered to be advertising information. The registry entries themselves are not added to the package.

- **Convert registry entries and retain registry information**
  Try to create a package that is accurate and supports advertising. SetupCapture will convert registry entries to advertising information, but it will also leave them as registry entries. This means that the installation instructions for these items exist twice in the Windows Installer package, but in different tables. This option might cause the application to constantly prompt the end user to perform a repair installation.

- **Capture Methodology**
  SetupCapture can use different methods for capturing the information from an installation.

  See *Selecting the Capture Methodology* on page 224.

- **Windows Installer Template**
  (SetupCapture Configuration only.)

  This field lists the templates in the Windows Installer Editor\Templates directory. Select the template to use as a basis for captured Windows Installer packages.

- **WiseScript Template**
  (SetupCapture Configuration only.)

  (Not available in Standard Edition.)

  This field lists the templates in the WiseScript Editor\Templates directory. Select the template to use as a basis for captured WiseScript packages.

- **Installation Changes Report**
  Specify whether to generate a report after each SetupCapture, and the report format. The items in the report comprise the repackaged installation. The report has the same file name and is created in the same directory as the repackaged installation created by SetupCapture.

  This option is unrelated to the View Report button on the SetupCapture Inclusions/Exclusions dialog boxes, which generates a temporary report regardless of what you select here.

### Setting Directories to Watch

Setting directories to watch is an important part of using SetupCapture. The directories you specify are scanned for changes, and the changes are incorporated into the repackaged installation.

The directories you specify are also used by the exclusion list building process.

See *Building an Exclusion List Automatically* on page 210.

SmartMonitor monitors all local directories, regardless of what is specified in Directories to Watch.

**Deciding which directories to watch**

At a minimum, specify \Program Files and \Windows or \Winnt. These are the most common directories in which changes occur during installation of applications.
Capturing Applications

The recommended setting is the root drive (C:\) with **Include Sub-directories** turned on, which means that the entire hard drive will be watched for changes. This provides the most complete record of file changes by capturing all changes made on the system. However, the problem with watching the entire drive is that you might capture changes that have nothing to do with installation, those that occur in the normal course of operating a computer. To avoid capturing irrelevant changes, you can build an exclusion list in SetupCapture Configuration.

See **Exclusion List Guidelines** on page 209.

**Note**

Because changing files on the root drive of the destination computer can have undesirable results, changes to the top level of the root drive are not captured even if you specify the root (C:\).

To override this and capture the root, mark **Allow root to be watched during capture** on the General Settings dialog box of SetupCapture Configuration.

**To set directories to watch**

1. Do one of the following:
   - Run SetupCapture Configuration and proceed until the Directories to Watch page appears.
   - On the Welcome page of SetupCapture, click Settings and click the Directories to Watch tab. If the Settings button is unavailable, you might not have permission to change settings.
     See **Setting SetupCapture Configuration Security** on page 44.

2. Click Add.
   The Select Directory dialog box appears.

3. Select a directory and click OK to add it to the list of watched directories. To watch all the subdirectories of the directory also, mark **Include Sub-Directories**. Click OK.

**Exclusion List Guidelines**

SetupCapture and SOE Snapshot use an exclusion list to determine which items on a computer to ignore. A basic exclusion list that contains operating system-related files is in the default configuration files in the share point directory and on the local computer. The basic exclusion list appears on the exclusion dialog boxes that appear in SetupCapture Configuration, SetupCapture, and SOE Snapshot.

You can exclude the following items: files, directories, files based on wildcards, registry values, and registry keys. Items that you exclude from SetupCapture are ignored for changes. Items that you exclude from SOE Snapshot are not recorded as part of the standard operating environment.

Generally, you should exclude:

- Anything that is temporary.
- Anything that is likely to be different based on the end user or hardware.
- Anything that is not applicable to Windows Installer technology.
  (Windows Installer packages only.)
Capturing Applications

- Changes that occur during installation that are unrelated to the actual installation. Example: If the installation displays a readme file in Internet Explorer, then several changes are made in the Internet Explorer directory that are the result of the readme file being opened, but are unrelated to the actual installation.
- (SOE Snapshot.) Items that you know are not part of the standard operating environment. Example: the C:\Temp directory.
- Changes made to certain system files during restart that are unrelated to the installation.
- Changed files that result from startup processes and services that run if the installation restarts.
- Files added to Temp directories after installation. These files are not usually meant to be part of the installation.
- Most Recently Used (MRU) lists in the registry that result from common tasks such as opening a text file or Word document.

See also:

Building an Exclusion List Automatically on page 210
Setting File and Folder Exclusions on page 211
Setting Registry Exclusions on page 214
Setting INI File Exclusions on page 215

Building an Exclusion List Automatically

On the Build Exclusion List page, you decide whether to build an exclusion list automatically, using snapshot comparison technology. An exclusion list is a list of files, folders, and registry keys and values to be ignored by SetupCapture or SOE Snapshot.


The exclusion list you build is stored in the current configuration file. You select the configuration file from SetupCapture, SOE Snapshot, or SetupCapture Configuration.

To build an exclusion list automatically

1. Close all applications except Wise Package Studio.
2. Run SetupCapture Configuration.
   See Configuring Settings in SetupCapture Configuration on page 203.
3. On the Directories to Watch page, specify the directories to be scanned.
4. On the Build Exclusion List page, mark Automatically build exclusion list (Recommended) and click Next.
   The Begin Exclusion Capture page appears.
5. Click Next to begin scanning. SetupCapture Configuration scans the registry and the directories specified in Directories to Watch.
   The Run Applications / Reboot page appears.
6. Run any applications that might be executed by an installation (examples: Internet Explorer, Notepad) and then restart the computer to ensure a complete exclusion list.
Leave the Run Applications / Reboot page open as you restart. Do not close it or click Cancel.

7. After restart, SetupCapture Configuration reopens to the Run Applications / Reboot page. Click Next.

   The End Exclusion Capture page appears.

8. Click Next. SetupCapture rescans the computer. Files and registry entries that changed while you ran applications and restarted are added to the exclusion list.

9. Subsequent dialog boxes in SetupCapture Configuration let you view and edit the exclusion list that was built. See:

   Setting File and Folder Exclusions

   Setting Registry Exclusions on page 214

   Setting INI File Exclusions on page 215

### Setting File and Folder Exclusions

You can specify files and folders to be ignored by SetupCapture or SOE Snapshot, or when building an exclusion list in SetupCapture Configuration. Set file and folder exclusions from SetupCapture Configuration, SetupCapture, or SOE Snapshot.

SetupCapture and SOE Snapshot automatically ignore certain system files.

See Files and Registry Entries Ignored During Captures on page 247.

You can exclude the following items:

- **A file**
  - See Setting a File to Be Excluded on page 211.
- **A directory**
  - You can exclude the files at the top level of the directory only, or all contents of the directory and its subdirectories.
  - See Setting a Directory to Be Excluded on page 212.
- **Files in a directory based on a wildcard**
  - Example: *.tmp for all temporary files.
  - See Setting a File to Be Excluded Based on a Wildcard on page 213.

See also:

Exclusion List Guidelines on page 209

### Setting a File to Be Excluded

You can specify files to be ignored by SetupCapture or SOE Snapshot, or when building an exclusion list in SetupCapture Configuration.

1. Do one of the following:

   - Run SetupCapture Configuration and proceed until the File and Folder Exclusions dialog box appears.
   - On the Welcome page of SetupCapture or SOE Snapshot, click Settings and click the File and Folder Exclusions tab. If the Settings button is unavailable, you might not have permission to change settings.
Capturing Applications

See *Setting SetupCapture Configuration Security* on page 44.

File and Folder Exclusions might already contain entries. Items with a question mark icon were added by the exclusion list building process. Items without a question mark icon were already in the exclusion list.

2. Click Add.

The File Exclude dialog box appears.

3. Complete the dialog box:
   - In **File/Wildcard**, specify a file.
   - **Directory** is pre-filled when you specify a file. You can use environment variables surrounded by percent signs (%) to specify paths. To exclude the file or wildcard for all local drives, leave this field blank.
   - To ignore the specified file in all subdirectories of the directory you specified, mark **Exclude Sub-Directories**. However, this still scans the excluded subdirectories so that the items that were excluded can be reported on the SetupCapture Exclusions page. To actually skip scanning of the subdirectories, also mark **Do Not Scan this directory and subdirectories**.
   - Click OK to return to the SetupCapture Configuration dialog.

Items that you specify will be ignored if they change during a capture while this configuration file is in effect. To edit an exclusion, double-click it in the list.

See also:

*Exclusion List Guidelines* on page 209

**Setting a Directory to Be Excluded**

You can set directories to be ignored by SetupCapture or SOE Snapshot.

Example: If a directory contains several excluded files, consider excluding the entire directory.

**To set a directory to be excluded**

1. Do one of the following:
   - Run SetupCapture Configuration and proceed until the File and Folder Exclusions dialog box appears.
   - On the Welcome page of SetupCapture or SOE Snapshot, click Settings and click the File and Folder Exclusions tab. If the Settings button is unavailable, you might not have permission to change settings.

See *Setting SetupCapture Configuration Security* on page 44.

File and Folder Exclusions might already contain entries. Items with a question mark icon were added by the exclusion list building process. Items without a question mark icon were already in the exclusion list.

2. Click Add.

The File Exclude dialog box appears.

3. Complete the dialog box:
In **Directory**, specify a directory. This causes SetupCapture to ignore files in the top level of this directory. You can use environment variables surrounded by percent signs (%) to specify paths. If you specify a path that contains user-specific data, a variable is inserted in place of the user-specific data.

To ignore the specified file in all subdirectories of the directory you specified, mark **Exclude Sub-Directories**. However, this still scans the excluded subdirectories so that the items that were excluded can be reported on the SetupCapture Exclusions page.

To actually skip scanning of the subdirectories, also mark **Do Not Scan this directory and subdirectories**.

Click OK to return to the SetupCapture Configuration dialog box.

4. If the exclusion list already contains files that are in the directory you specified, you are prompted to remove the redundant entries. Click Yes.

Items that you specify will be ignored if they change during a capture while this configuration file is in effect. To edit an exclusion, double-click it in the list.

### Setting a File to Be Excluded Based on a Wildcard

You can set files to be ignored based on wildcards. SetupCapture and SOE Snapshot ignore changes to all files that match the wildcard criteria within a particular directory.

**To set a file to be excluded based on a wildcard**

1. Do one of the following:
   - Run SetupCapture Configuration and proceed until the File and Folder Exclusions dialog box appears.
   - On the Welcome page of SetupCapture or SOE Snapshot, click Settings and click the File and Folder Exclusions tab. If the Settings button is unavailable, you might not have permission to change settings.

      See [Setting SetupCapture Configuration Security](#) on page 44.

     File and Folder Exclusions might already contain entries. Items with a question mark icon were added by the exclusion list building process. Items without a question mark icon were already in the exclusion list.

2. Click Add.

   The File Exclude dialog box appears.

3. Complete the dialog box:

   - In **Directory**, specify a directory. The wildcard you set will apply to files in this directory. You can use environment variables surrounded by percent signs (%) to specify paths. (Example: %TEMP%\*.tmp or %WinDir%\Exclude.dll)

   - In **File/Wildcard**, enter a wildcard. If the file or directory is under a user profile, the user profile name is replaced with a variable that always represents the current user profile name.

      See [Converting User-Specific Files to Generic User Files](#).

   - To ignore the specified file in all subdirectories of the directory you specified, mark **Exclude Sub-Directories**. However, this still scans the excluded subdirectories so that the items that were excluded can be reported on the SetupCapture Exclusions page.
To actually skip scanning of the subdirectories, also mark **Do Not Scan this directory and subdirectories**.

- Click OK to return to the SetupCapture Configuration dialog box.

Items that you specify will be ignored if they change during a capture while this configuration file is in effect. To edit an exclusion, double-click it in the list.

### Converting User-Specific Files to Generic User Files

If you specify file, folder, or registry exclusions that have user-specific data in their paths, the following prompt appears:

> One or more file exclusions are located under your current user profile. Would you like to change these file exclusions so that they apply to any currently logged-in user profile?

If you click Yes in this prompt, a variable is inserted in place of user-specific data in the exclusion list. This lets the SetupCapture configuration file work on any computer it is transferred to.

**Example:**

Suppose the following file is in the exclusion list:

```
C:\Documents and Settings\Mariem\Cookies\Index.dat
```

That path won’t exist on anyone else’s computer, because a user name is part of the path. If you click Yes in the prompt, the entry above is changed to:

```
Current_User_Profile\Cookies\Index.dat
```

`Current_User_Profile` will be replaced with the user profile folder for the computer that is running SetupCapture.

You edit the exclusion list from SetupCapture Configuration, SetupCapture, or SOE Snapshot.

See [Setting File and Folder Exclusions](#) on page 211 and [Setting Registry Exclusions](#) on page 214.

### Setting Registry Exclusions

You can specify registry keys and values to be ignored by SetupCapture or SOE Snapshot.

1. Do one of the following:
   - Run SetupCapture Configuration and proceed until the Registry Exclusions dialog box appears.
   - On the Welcome page of SetupCapture or SOE Snapshot, click Settings and click the Registry Exclusions tab. If the Settings button is unavailable, you might not have permission to change settings.
     
     See [Setting SetupCapture Configuration Security](#) on page 44.

Registry Exclusions might already contain entries. Items with a question mark icon were added by the exclusion list building process. Items without a question mark icon were already in the exclusion list.

2. Click Add.

The Exclude Registry Key dialog box appears.
3. In the left pane, select a registry key.
   - To exclude a particular value, click the value name in the right pane.
   - To exclude an entire registry key, click the `<ignore entire subtree>` entry in the right pane. You might do this if one registry key contains several excluded values.

4. Click OK to return to the SetupCapture Configuration dialog box.
   
   If you specified an entire registry key, and the exclusion list already contains values in that key, you are prompted to remove the redundant entries. Click Yes.

   Items that you specify will be ignored if they change during a capture while this configuration file is in effect. To edit an exclusion, double-click it in the list.

See also:

*Exclusion List Guidelines* on page 209

**Setting INI File Exclusions**

You can specify .INI files or parts of .INI files to be ignored by SetupCapture and SOE Snapshot.

**To set INI file exclusions**

1. Do one of the following:
   - Run SetupCapture Configuration and proceed until the INI File Exclusions dialog box appears.
   - On the Welcome page of SetupCapture or SOE Snapshot, click Settings and click the INI File Exclusions tab. If the Settings button is unavailable, you might not have permission to change settings.
     
     See *Setting SetupCapture Configuration Security* on page 44.

2. Click Add.
   
   The Exclude INI File, Section, or Entry dialog box appears.

3. In **INI File Name**, enter the exact name of the .INI file, including the extension .INI.

4. Complete the dialog box:
   - To exclude an entire .INI file, enter * in **Section/Wildcard**.
   - To exclude a section of an .INI file, enter the section name (with or without brackets) in **Section/Wildcard**. You can use wildcards. Leave **Entry/Wildcard** blank.
   - To exclude an entry in an .INI file, enter the section name (with or without brackets) in **Section/Wildcard**. Then enter the entry name in **Entry/Wildcard**. You can use wildcards in both fields.
   - Click OK to return to the SetupCapture Configuration dialog box.

   Items that you specify will be ignored if they change during a capture while this configuration file is in effect. To edit an exclusion, double-click it in the list.

See also:
SetupCapture

SetupCapture® records all the changes performed by an installation and saves that information to a new Windows Installer, WiseScript, or virtual software package. (The ability to create a WiseScript or virtual software package is not available in Standard Edition.) For information on saving a captured application as a virtual software package, see About SetupCapture in the Virtual Package Editor Help. SetupCapture can also capture the first use changes that an application makes to a computer.

SetupCapture uses several capture methods:

- **Virtual Capture™**, which lets you capture installations on a non-clean machine. The application you capture is installed into a simulated computer directory structure and its registry entries are installed into a simulated registry. (Not available in Standard Edition.)
- **SmartMonitor™**, in which SetupCapture watches the installation and records the changes the installation performs as they happen.
- **Snapshot**, in which SetupCapture scans the computer before and after the installation and records the differences between the two scans.
- A combination of SmartMonitor and Snapshot.

If you have the Software Virtualization Agent installed, you can use the SmartMonitor and snapshot methods to capture the installation in a virtual software layer.

See Capturing an Installation in a Virtual Software Layer on page 219.

Configuration files control how SetupCapture works. You select the configuration file when you run SetupCapture. To create and edit configuration files, use SetupCapture Configuration.


It is extremely important to create a robust exclusion list before using SetupCapture. The exclusion list is contained in the configuration file. It consists of files, directories, files based on wildcards, registry values, and registry keys that should be ignored by SetupCapture.

**Note**

To capture an application on a computer that is running Windows 95/98/NT 4.0, use Wise Web Capture instead of SetupCapture.

See Capturing With Wise Web Capture on page 245.

Guidelines for Capturing an Installation

Before using SetupCapture for the first time, read the following guidelines to learn how SetupCapture works. For guidelines on saving a captured application as a virtual software package, see Guidelines for Capturing an Installation in the Virtual Package Editor Help.

- Run SetupCapture on a clean machine.
  
  See Setting Up a Clean Machine on page 218.
In the Professional Edition, you can use Virtual Capture to simulate a clean machine. Before using Virtual Capture, read the guidelines in *Using SetupCapture With Virtual Capture* on page 237.

- You can capture an installation in a virtual software layer and then delete or deactivate the layer and restore the computer to its original state. See *Capturing an Installation in a Virtual Software Layer* on page 219.

- During a capture, SetupCapture tries to convert computer- and user-specific data in the registry to generic data that will work on any computer. It does this by searching for standard paths (example: C:\Winnt) and replacing them with Windows Installer properties (example: [WindowsFolder]). Part of this process includes searching for the computer name and currently logged-on user name. To make the search for computer and user names as accurate as possible, make sure the computer name and user name on the capture computer are set to unique names four or more characters in length. Avoid having the user name or computer name set to any common file or folder names. An example of a unique user name is: repackage-1-user.

- Before you run SetupCapture, exit all other applications, including background services or applications. (Example: Norton AntiVirus.)

- If you capture to a WiseScript package, be aware that for display purposes, the registry entries are split into multiple Edit Registry actions. (Not available in Standard Edition.)

- During SetupCapture, changes to an .INI file are recorded as changes to an .INI file only if the .INI file follows standard .INI file format. Otherwise, the changes are recorded as a file change.

- Do not capture an .MSI-based installation. Instead, open the .MSI directly in Windows Installer Editor. To customize it for specific workgroups, create a transform. See *Creating a Transform Based on an Existing .MSI* in the Windows Installer Editor Help.

- You must be able to run the original installation to repackage it with SetupCapture. Example: If the installation requires a serial number, you must have the serial number.

- SetupCapture does not monitor any internal logic within the installation and it does not replicate the user interface of the original installation.

- SetupCapture creates a separate feature for each .EXE that’s installed that has a shortcut. Isolating .EXE components into features results in more efficient repairs, because if there is a problem with a component, only the problem component and the .EXE are reinstalled instead of the entire feature containing the problem component. (Windows Installer packages only.)

- To capture an uninstall, you must mark *Include files deleted during capture* and *Include registry keys deleted during capture* in SetupCapture Configuration General Settings. In Windows Installer Editor, deleted items are located in the RemoveFile and RemoveRegistry tables in Setup Editor > Tables tab. In WiseScript Package Editor, deleted items are located in Delete File(s) or Edit Registry statements in the script.

- Registry keys that define an environment variable are converted to an environment variable in the repackaged installation.
Look for environment variables in the Features or Components tabs of Setup Editor.
(Windows Installer packages only.)

Setting Up a Clean Machine

A clean machine is a computer containing only the operating system and its service packs. A baseline machine is a computer with the operating system, basic system software, and additional applications that are installed on every computer in your organization. Example: antivirus software.

Recommendations

- Run all SetupCaptures on a clean machine. This makes repackaged installations more resilient by making them less dependent on the existence of other applications.
- If you have the Software Virtualization Agent installed, you can capture an installation in a virtual software layer. You can then delete or deactivate this virtual layer to restore the clean machine to its original state.
  
  See Capturing an Installation in a Virtual Software Layer on page 219.
- If you use Virtual Capture, you don’t need to run SetupCapture on a clean machine, but you do need to create a Virtual OS of a clean machine.
  
  See Creating a Virtual OS on page 239.
- In general, run all SOE Snapshots on a baseline machine. To capture only the operating system with SOE Snapshot, use a clean machine.

If you run SetupCapture on a baseline machine, all the repackaging and conflict resolution work you do becomes suspect if you upgrade any of the additional applications. Example: If you upgrade from version 3.0 to version 4.0 of your antivirus software, system .DLLs or other files might change. As a result, items that did not conflict before might conflict now.

Installing Wise Package Studio on a Clean Machine

Because Wise Package Studio minimizes the changes to a computer, you generally can install Wise Package Studio on a clean machine without affecting the quality of the repackaged applications. However, Wise Package Studio does install MDAC. If you don’t want it on your clean machine, do one of the following:

- Use Wise Web Capture to perform captures from a browser without installing any additional software.
  
  See Capturing With Wise Web Capture on page 245.
- Use Virtual Capture to perform captures on a non-clean machine.
  
  See Using SetupCapture With Virtual Capture on page 237.

How to Set Up a Clean Machine

1. Perform a clean installation of an operating system on the computer you use for capturing applications.

2. Install any service packs.

3. Install Windows Installer if it is not included with the operating system.

To replicate the clean machine quickly and easily, use a drive imaging tool or if you captured the installation into a virtual layer, delete or deactivate the virtual layer.
Capturing an Installation in a Virtual Software Layer

If you have the Software Virtualization Agent installed, you can capture an installation in a virtual software layer. You can use this option if you are saving the installation as a Windows Installer or WiseScript package. All changes made to the computer when you capture the installation are put into the layer. You can then use the Symantec SVS applet to delete or deactivate the layer and restore the computer to its original state.

See About the Altiris SVS Applet in the Virtual Package Editor Help.

You capture installations into a layer by marking **Use SVS to capture into a layer** on the Execute Installation page.

Capturing an installation to a virtual software layer is different from using Virtual Capture. Virtual Capture lets you achieve clean machine results while capturing on a non-clean machine. This eliminates the need to reimage your computer between each SetupCapture. Capturing an application to a virtual software layer also eliminates the need to reimage your computer between each SetupCapture, but you don’t have to create a Virtual OS. The only requirement for capturing applications into a virtual software layer is the installation of the Software Virtualization Agent, which is installed with Wise Package Studio.

For general information about the Software Virtualization Solution, see Integration with Software Virtualization Solution on page 29.

**Note**

If the output of the capture is an uncompiled file (.WSI or .WSE), either import the package into Software Manager or compile it before you delete the virtual layer. If you delete the layer first, you lose the package's source files.

Capturing an Installation

➢ **Requires Windows Installer.**

Windows Installer must be installed on the computer on which you run SetupCapture.

SetupCapture® records all the changes performed by an installation and saves that information to a new Windows Installer, WiseScript, or virtual software package. You can further customize the repackaged installation in Windows Installer Editor, WiseScript Package Editor, or Virtual Package Editor. (The ability to create a WiseScript or virtual software package is not available in Standard Edition.)

For information on saving a captured application as a virtual software package, see About SetupCapture in the Virtual Package Editor Help.

**To capture an installation**

1. (Not available in Standard Edition.) If you plan to use Virtual Capture, first create a Virtual OS.
   
   See Creating a Virtual OS on page 239.

2. Exit all other applications so that changes they make are not captured.

3. Do one of the following:
   
   ■ On the Projects tab, click the Run link to the right of the task or tool associated with SetupCapture. The package that is created will be saved with the default
project name. This tool might skip pages or populate fields based on command-line options defined in Process Templates Setup.

- On the Tools tab, double-click SetupCapture.

4. If the SetupCapture Type page appears, select the type of capture to perform.

  - **SetupCapture**
    This performs a typical SetupCapture, which captures the changes made by an installation.

  - **First Use Settings**
    This creates a transform file that captures the changes made by an .MSI-based application the first time it is used after being installed. You can apply this transform to the base .MSI to pre-apply the first use settings. If you choose First Use Settings, do not use this procedure; use SetupCapture instead.

    See Using SetupCapture to Capture First Use Settings on page 240.

5. If the Specify Target Installation File page appears, specify what to do with the results of the SetupCapture and then click Next. (For details on other options, see Specifying the Installation File on page 223.)

  - **Target Installation**
    Specify the full path of a new or existing .MSI, .WSI, or .WSE file in which to store the SetupCapture results. (In the Standard Edition, you cannot specify a .WSE.) For a new installation, specify a local directory. To append to an existing installation, specify the path to the existing file.

  - **Add/Update Resources in Existing Installation**
    If you specified an existing installation in Target Installation, you can mark this check box to append or update the resources from the capture in the existing installation instead of overwriting the existing installation.

    **Note**
    If you previously ran SetupCapture from the Projects tab, an installation file already exists. If you run it again, you might be prompted to create a new file or append to the existing file. If this prompt does not appear, your original file will be overwritten.

The Welcome page appears.

6. On the Welcome page:

  - To change the configuration settings for this capture without changing the current configuration file, select the **Do not change current configuration file** option.
    This option is enabled only if the user has permission to use the SetupCapture Configuration tool. If the Settings button is disabled, selecting this option enables it. The user can then modify the settings for the current capture.

  - To select a different configuration file, click Change. This session of SetupCapture will use the settings in the specified configuration file. If you select the **Do not change current configuration file** option, this option is disabled.
    See Selecting the Configuration File on page 205.

  - To change the configuration settings for the current SetupCapture, click Settings. By default, anything you change on all tabs of this dialog box is saved to the current configuration file and affects future captures. If you don’t want to
change the configuration file, mark the check box **Do not change current configuration file**.

See:

- Setting General Settings on page 205
- Setting Directories to Watch on page 208
- Setting File and Folder Exclusions on page 211
- Setting Registry Exclusions on page 214
- Setting INI File Exclusions on page 215

If the Change button or the Settings button is unavailable, you might not have permission to change settings.

See Setting SetupCapture Configuration Security on page 44.

7. Click Next on the Welcome page.

8. On the Capture Methodology page, select a method for capturing installations and click Next.

See Selecting the Capture Methodology on page 224.

If you selected the snapshot method, the Initial Scan page might appear. Specify whether to rerun the initial scan or use the initial scan that was created previously.

See Using a Previous Scan on page 227.

If you selected Virtual Capture, the Virtual OS File Selection page appears.

9. If the Virtual OS File Selection page appears, specify a Virtual OS to use and click Next.

You must have run the Virtual OS Creation utility on the current computer or another computer.

See Creating a Virtual OS on page 239.

- **Use the existing Virtual OS file**
  
  Mark this if you are working in a clean build environment and you previously ran the Virtual OS Creation utility on the current computer. This option is unavailable if no Virtual OS is found on this computer.

- **Use a different Virtual OS file from the share point directory or network**
  
  Mark this if you previously ran the Virtual OS Creation utility on another computer and the resulting .WOS file is available in the share point directory or in a network directory. Then specify the .WOS file to use.

10. Click Next.

The Begin Installation Capture page appears.

11. Click Next.

What happens next depends on the capture methodology you chose:

- **Virtual Capture:** If you selected the existing Virtual OS file, the Virtual OS is cleaned. This means that installed remnants from the last capture are removed from the Virtual OS directory and registry structure. If you selected a .WOS file, it is expanded onto your computer to form a Virtual OS directory and registry structure. This process takes several minutes and requires a substantial amount of free disk space.
Capturing Applications

- SmartMonitor: Monitoring begins.
- Snapshot: Your computer is scanned, unless you are using the previous initial scan. Do not work on the computer during the scan.

The Execute Installation page appears.

12. On the Execute Installation page, do the following:
   a. Specify the full path of the installation executable in .EXE Name.
   b. To run the installation with command-line options, enter them in Command Line.
   c. To capture the installation into a virtual software layer, mark Use SVS to capture into a layer.
      
      See Capturing an Installation in a Virtual Software Layer on page 219.
   d. To run only one installation, click Next and run the installation. To run multiple installations, click Execute and then repeat the process for each installation. Click Next only after you finish all installations.
      
      See Executing Installations to Be Captured on page 228.

   **Note**
   If you are using Virtual Capture, it is very important to run installations from the Execute or Next button on this page. If you run installations outside this page, SetupCapture will not capture them.

   The End Installation Capture page appears.

13. Click Next.

   (Standard Edition.) The Finish page appears. Skip the next steps that describe the Inclusions and Exclusions pages.

   (Professional or Enterprise Edition.) The SetupCapture Inclusions page appears. The SetupCapture Inclusions page displays the items that will be added to the repackaged installation. These items represent changes that were detected during scanning or monitoring.

14. On the SetupCapture Inclusions page, to remove an item from the repackaged installation, select it in the list and click Exclude. The Exclude Globally button causes it to be excluded from future captures. To display another type of item, select the type from Inclusion Type.

   See Editing SetupCapture Inclusions on page 228.

15. Click Next on the SetupCapture Inclusions page.

   The SetupCapture Exclusions page appears. It displays all the changed items that are excluded from the repackaged installation based on the exclusion list.

16. On the SetupCapture Exclusions page, to add an item back into the repackaged installation, select it in the list and click Include. To display another type of item, select the type from Inclusion Type.

   See Editing SetupCapture Exclusions on page 229.

The Finish page appears.
17. Enter summary information for the repackaged installation. This information will appear in the corresponding fields on the appropriate Installation Expert pages in Windows Installer Editor or WiseScript Package Editor.

See *Finishing SetupCapture* on page 230.

18. (Windows Installer packages only.) You can save the results from this SetupCapture to a specific feature. Specify an existing feature or click New to create a new feature.

See *Configuring the Installation as a New Feature* on page 233.

19. Click Finish.

- If files in the captured installation depend on certain merge modules, the Download Redistributables wizard appears. Use the Wise Web Site option to download those merge modules, which should be pre-selected.
  
  See *Downloading Redistributable Files* in the Windows Installer Editor Help.

- If a file that is part of a merge module is added, the Files in Merge Modules dialog box appears. It prompts you to add the merge module and, if necessary, download it.
  
  See *Adding Merge Modules Instead of Files* on page 232.

If you ran SetupCapture from the Projects tab, the repackaged installation is saved in the project directory. If you ran SetupCapture from the Tools tab, the repackaged installation is saved to the file name and location you specified at the beginning of SetupCapture.

### Specifying the Installation File

On the Specify Target Installation File page, you specify the repackaged installation that results from the SetupCapture and how to handle source files. This page appears during SetupCapture. It might not appear when you run SetupCapture from a task in the Workbench Projects tab.

See *Capturing an Installation* on page 219.

- **Target Installation**
  
  Specify the full path of a new or existing .MSI, .WSI, or .WVP file in which to store the SetupCapture results. In the Professional Edition, you also can specify a .WSE.

- **Add/Update Resources in Existing Installation**
  
  If you specified an existing installation in Target Installation, you can mark this check box to append or update the resources from the capture in the existing installation instead of overwriting the existing installation.

- **Leave Source Files in Original Location**
  
  For a .WSI, .MSI, or .WSE file, mark this to have all the source files in the repackaged installation referenced from their installed location. (Example: If you installed Sample on your C drive during SetupCapture, the source file paths would be: C:\Program Files\Sample\Sample.exe.)

  For a .WVP file, you must specify a directory where the .WVP source files are to be copied either on this dialog box or on the Finish dialog box.

- **Copy Source Files During Installation Save**
  
  Mark this to make a copy of all the installed files in the Destination Directory that you specify. The files are copied near the end of the SetupCapture. A separate
installation file is created in the directory you specify and it references the copied source files.

For a .WSI, .MSI, or .WSE, this lets you immediately reimage the test computer and move the entire installation to another computer. The file that was specified in Target Installation is still created, but it references the installed source files, not the copied source files.

- **Store Source File Pathnames as Relative Pathnames**
  Mark this to make the captured installation more mobile by making the source file paths relative to the location of the installation file. If you mark this option, place the resulting .MSI, .WSI, .WSE, or .WVP in the same or a related directory to the destination directory (above).

### Configuring SetupCapture

On the Welcome page in SetupCapture, you can:

- Mark the **Do not change current configuration file** option. If you select this option, any changes you make to the configuration settings apply only to the current capture. The SetupCapture configuration file is not changed.
  This option is enabled only if the user has permission to use the SetupCapture Configuration tool. If the Settings button is disabled, selecting this option enables it. The user can then modify the settings for the current capture.

- Change the configuration file that controls how SetupCapture works. To use a different configuration than the one listed on the Welcome page, click Change, which opens the Configuration File dialog box. If you select the **Do not change current configuration file** option, this option is disabled. See [Selecting the Configuration File](#) for details.

- Change or review the settings in the SetupCapture configuration file. Click Settings to open the SetupCapture Configuration dialog box. By default, anything you change on all tabs of this dialog box is saved to the current configuration file and affects future captures. If you don’t want to change the configuration file, mark the check box **Do not change current configuration file**.
  For details on the SetupCapture Configuration dialog box, see:
  - [Setting General Settings](#) on page 205
  - [Setting Directories to Watch](#) on page 208
  - [Setting File and Folder Exclusions](#) on page 211
  - [Setting Registry Exclusions](#) on page 214
  - [Setting INI File Exclusions](#) on page 215

If the Change button or the Settings button is unavailable, you might not have permission to change settings.

See [Setting SetupCapture Configuration Security](#) on page 44.

### Selecting the Capture Methodology

SetupCapture can use different methods for capturing the information from an installation. You select the capture method from the Capture Methodology page in SetupCapture or from an option on the General Settings dialog box of SetupCapture Configuration.
Capturing Applications

Note
A security setting in Windows Vista or later operating system prevents the Virtual Capture and SmartMonitor methods from working. When you try to use either method, a prompt appears and provides the option to disable the security restriction. If you choose to disable the security restriction, the following registry setting is set:

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Windows>LoadAppInit_DLLs=1

The initial default for this setting is 0. If you disable this restriction, your computer’s vulnerability to malicious attack is increased. However, if you run SetupCapture in a testing environment, the increased vulnerability might not be a critical issue.

- **Virtual Capture™**
  Virtual Capture lets you capture installations on a non-clean machine. The application you capture is installed into a simulated computer directory structure and its registry entries are installed into a simulated registry. To use Virtual Capture, you must first create a Virtual OS, being sure to meet the Virtual Capture requirements.

  See *Using SetupCapture With Virtual Capture* on page 237.

Note
Virtual Capture is not available if you chose First Use Settings on the SetupCapture Type page.

- **SmartMonitor™**
  SetupCapture monitors and records the installation’s operations as they happen. This method is faster than snapshot comparisons, because it doesn’t require a time-consuming scan of the computer.

  SmartMonitor records the following operations:

  - Copying, moving, deleting, or opening a file
  - Replacing files even if they are the same size, modification date, and version
  - Creating or removing a directory
  - Creating, starting, stopping, or deleting a service
  - Setting or deleting a registry value, creating or deleting a registry key.
  - Overwriting existing registry keys with the same value
  - Installing ODBC drivers or configuring ODBC data sources
  - Changing .INI files regardless of their location

Note
SmartMonitor monitors all local directories, regardless of what is specified in Directories to Watch.

- **Snapshot**
  SetupCapture scans the computer before installation occurs, then you perform the installation, then SetupCapture rescans the computer. SetupCapture records the differences between the two scans and adds them to the repackaged installation.

  The disadvantages of this method are:

  - It does not capture the replacement of files if the files are the same size, modification date, and version.
Capturing Applications

- It does not capture overwriting of existing registry keys.
- .INI file changes are handled differently—if an .INI file is in the Windows directory, changes to it are recorded as an .INI file change. If an .INI file is outside the Windows directory, the entire .INI file is added instead of just editing the file.

- **Use SmartMonitor in conjunction with Snapshot**
  Merge the results of both methods. If the results of the two methods don’t match, the differences between the methods are added to the repackaged installation as well. This provides the most accurate and complete representation of the captured installation.

See also:

*Capturing an Installation* on page 219

*Setting General Settings* on page 205

### Selecting a Virtual OS File

> Not available in Standard Edition.

On the Virtual OS File Selection page, specify a Virtual OS to use. This page appears during SetupCapture, but only if you use Virtual Capture.

See *Capturing an Installation* on page 219.

You must have run the Virtual OS Creation utility on the current computer or another computer.

See *Creating a Virtual OS* on page 239.

- **Use the existing Virtual OS file**
  Mark this if you previously ran the Virtual OS File Creation utility on the current computer. When you click Next, the Virtual OS is cleaned. This means that installed remnants from the last capture are removed from the Virtual OS directory and registry structure.

  This option is unavailable if no Virtual OS is found on this computer.

- **Use a different Virtual OS file from share point directory or network**
  Mark this if you previously ran the Virtual OS Creation utility on another computer and the resulting file is available in the share point directory or in a network directory. Then specify the .WOS file to use. When you click Next, the Virtual OS file you specified is expanded onto your computer as a Virtual OS directory and registry structure. This process takes several minutes and requires a substantial amount of free disk space.

### Beginning the SetupCapture

On the Begin Installation Capture page, you start the capture process. This page appears during SetupCapture.

See *Capturing an Installation* on page 219.

Before you begin the SetupCapture, close all applications except Wise Package Studio and SetupCapture.

Click Next to begin the capture. The capture method determines what happens.
**Virtual Capture™**

If you selected the existing Virtual OS, the Virtual OS is cleaned. This means that installed remnants from the last capture are removed from the Virtual OS directory and registry structure.

If you selected a .WOS file, it is expanded onto your computer as a Virtual OS directory and registry structure. This process takes several minutes and requires a substantial amount of free disk space.

See *Using SetupCapture With Virtual Capture* on page 237.

**SmartMonitor™**

SetupCapture starts a monitoring process that watches all local directories for changes.

**Snapshot**

SetupCapture performs an initial scan to determine the contents of the computer before you run the installation to be captured. It scans the directories specified in Directories to Watch and the registry. Depending on the contents of your computer, the initial scan can take several minutes. Do not work on the computer during the scan.

If you are using a previous initial scan, the scan is skipped.

**SmartMonitor in conjunction with Snapshot**

SetupCapture starts monitoring and scanning.

**Using a Previous Scan**

On the Initial Scan page, specify whether to rerun the initial scan or use the initial scan that was created previously.

This page appears during SetupCapture if you selected the snapshot method and one of the following is true:

- You previously started a SetupCapture but canceled before it completed. In that case, use the initial scan that was created previously.
- You previously performed a SetupCapture on this computer. Because it is recommended that you perform SetupCaptures on a clean machine, cancel this SetupCapture and reimage the computer or reinstall the operating system before proceeding. The exceptions are if you are capturing the results of an uninstall, in which case it is appropriate to use the previous scan, or if you are using Virtual Capture.
- You made an image of a drive after the initial scan was completed and use that image for all new SetupCaptures. In that case, use the initial scan that was created previously.

If you are running multiple SetupCaptures on a single computer, and if you use a drive imaging application, you can use the Initial Scan page to save time when you perform new SetupCaptures. On a clean machine, start SetupCapture and run it just long enough to perform an initial scan. Then cancel SetupCapture and make an image of the hard drive. Use that image to perform subsequent captures.

See also:

*Capturing an Installation* on page 219
Capturing Applications

Executing Installations to Be Captured

On the Execute Installation page, specify and run one or more installations. The installations are recorded and their changes are added to the repackaged installation that results from the capture. This page appears during SetupCapture.

See Capturing an Installation on page 219.

Do not capture an .MSI-based installation. Instead, open the .MSI in Windows Installer Editor.

Note
If you are using Virtual Capture, it is very important to run installations from the Execute or Next button on this page. If you run installations outside this page, SetupCapture will not capture them.

To run one or more installations

1. In .EXE Name, specify the full path of the installation executable.
2. (Optional.) In Command Line, specify command-line options to run with the installation.
3. (Optional.) To capture the application in a virtual layer, mark Capture the application in a virtual software layer.
   See Capturing an Installation in a Virtual Software Layer on page 219.
4. Click Execute.
   The installation starts. Run the installation, installing the product as you want it to be captured, and return to SetupCapture when the installation is finished.
5. To run additional installations, repeat the preceding steps for each installation. The option to capture to a layer is unavailable now because this option must be the same for each captured installation.
   All installations you run will be added to the repackaged installation.
6. When you finish running installations, click Next on the Execute Installation page.

Editing SetupCapture Inclusions

➢ Not available in Standard Edition.

The SetupCapture Inclusions page displays the items that will be added to the repackaged installation. Use this page to remove an item from the repackaged installation and to view a report of inclusions and exclusions. This page appears during SetupCapture.

See Capturing an Installation on page 219.

The icons to the left of the items indicate how the items are affected by the installation:

➢ Item is added by the installation.
➢ Item is changed or updated by the installation.
➢ Item is removed by the installation.
Using the SetupCapture Inclusions page:

- To display another type of item, select the type from **Inclusion Type**. Included items are separated into the following types: files, registry keys, .INI files, and shortcuts.

- To remove an item from the repackaged installation, select it in the list and click **Exclude**. This adds it to the exclusions list, which appears on the SetupCapture Exclusions page. You might want to remove an item if it is unrelated to the installation that was captured.

- To add an item to the permanent exclusion list for the current configuration file, select the item and click **Exclude Globally**. If the Exclude Globally button is unavailable, you might not have permission to change configuration settings. See **Setting SetupCapture Configuration Security** on page 44.

- To remove an item from the permanent exclusion list, run SetupCapture Configuration and edit the configuration file. See **SetupCapture Configuration** on page 202.

- To treat an .INI file as a regular file, select it in the .INI files list and click **Treat as File**. To revert to treating the file as an .INI file, select it in the Files list and click **Treat as INI**. This function is useful when you want to overwrite a copy of an .INI file on the destination computer.

  When SetupCapture detects a non-standard .INI file, it treats it as a regular file, listing it in the Files list of the SetupCapture Inclusions page and creating a single entry for the whole file in the Files table (rather than creating several entries in the IniFile table). You cannot move non-standard .INI files from the Files list of the SetupCapture Inclusions page to the .INI files list.

- If you are not using SmartMonitor, click **View Report**. An .HTM-formatted list of all file, .INI file, and registry inclusions and exclusions appears. The inclusions and exclusions are grouped by the type of system change.

  If you are using SmartMonitor:
  - Click **View Report** and select **Installation Changes** from the button menu. An .HTM-formatted list of all file, .INI file, and registry inclusions and exclusions appears. The inclusions and exclusions are grouped by the type of system change.
  - Click **View Report** and select **Installation Sequence** from the button menu. A .TXT-formatted report appears, listing sequence of actions that took place during installation.

When you finish editing inclusions, click **Next**.

**Editing SetupCapture Exclusions**

➤ **Not available in Standard Edition.**

The SetupCapture Exclusions page displays items that are excluded from the repackaged installation based on the exclusion list. These items might be excluded because you specified them on the SetupCapture Inclusions page, or because they match the exclusion criteria in the current configuration file. Use this page to add items that are currently set to be excluded back into the repackaged installation, and to view a report of inclusions and exclusions.

This page appears during SetupCapture.
Capturing Applications

See *Capturing an Installation* on page 219.

The icons to the left of the items indicate how the items are affected by the installation:

- Item is added by the installation.
- Item is changed or updated by the installation.
- Item is removed by the installation.

**Using the SetupCapture Exclusions page:**

- To display another type of item, select the type from **Exclusion Type**. Excluded items are separated into the following types: files, registry keys, .INI files, and shortcuts.

- To add an item back into the repackaged installation, select it in the list and click Include. You might want to add an item if it is part of the installation that was captured. If you previously globally excluded the item, and you include it here, it is included for the current SetupCapture, but will still be excluded for subsequent SetupCaptures.

- If you are not using SmartMonitor, click View Report. An .HTM-formatted list of all file, .INI file, and registry inclusions and exclusions appears. The inclusions and exclusions are grouped by the type of system change.

- If you are using SmartMonitor:
  - Click View Report and select Installation Changes from the button menu. An .HTM-formatted list of all file, .INI file, and registry inclusions and exclusions appears. The inclusions and exclusions are grouped by the type of system change.
  - Click View Report and select Installation Sequence from the button menu. A .TXT-formatted report appears, listing sequence of actions that took place during installation.

When you finish editing exclusions, click Next.

**Finishing SetupCapture**

On the SetupCapture Finish page, enter summary information for the repackaged installation. This information will appear in the corresponding fields on the appropriate Installation Expert pages in Windows Installer Editor or WiseScript Package Editor. In Windows Installer packages, you can place the captured installation resources in a new feature.

This page appears during SetupCapture.

See *Capturing an Installation* on page 219.

**How Entries Correspond to Fields in Installation Expert**

<table>
<thead>
<tr>
<th>Entry</th>
<th>Windows Installer Editor</th>
<th>WiseScript Package Editor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Field: <strong>Name</strong></td>
<td>Field: <strong>Installation Title</strong></td>
</tr>
<tr>
<td></td>
<td>Page: Product Details</td>
<td>Page: Product Details</td>
</tr>
</tbody>
</table>
Completing the Finish page

Enter information in the following fields:

- **Name**
  The name of the application or software package. The end user sees this name during installation of a Windows Installer or WiseScript package.

- **Version**
  The version number of the application or software package.

  Windows Installer uses this version number to identify the product when subsequent patches or upgrades are applied. For Windows Installer, it should be in the format AA.BB.CCCC.DDDD, where AA is the major version, BB is the minor version, CCCC is the build version, and DDDD is optional and ignored. It is stored as a string data type.

  In a WiseScript, this version number sets the version resource of the compiled setup file (.EXE).

- **Manufacturer**
  The manufacturer or publisher of the software.

- **Default Directory**
  (Windows Installer packages only.)

  During installation, this directory is displayed to the end user on the Destination Directory dialog box, and the end user can change the default location for the application. (The Destination Directory dialog box is called the Single Feature Destination dialog box in Windows Installer Editor.)

- **Destination Feature**
  (Windows Installer packages only.)

  To save results from this SetupCapture to a specific feature, either specify an existing feature or click New to create a new feature.

  See *Configuring the Installation as a New Feature* on page 233.

Click Finish to save the SetupCapture results.

**Warning**

Click the Finish button, not the close box, on the Finish page. If you click the close box, you lose the results of the capture.

If you ran SetupCapture from the Projects tab, the repackaged installation is saved in the project directory. If you ran SetupCapture from the Tools tab, the repackaged installation is saved to the file name and location you specified at the beginning of SetupCapture. To see the results of the SetupCapture, open the repackaged installation in Windows Installer Editor or WiseScript Package Editor.

One or more additional dialog boxes might appear at the end of SetupCapture:

<table>
<thead>
<tr>
<th>Entry</th>
<th>Windows Installer Editor</th>
<th>WiseScript Package Editor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>Field: <strong>Version</strong></td>
<td>Field: <strong>Installation Version</strong></td>
</tr>
<tr>
<td></td>
<td>Page: Product Details</td>
<td>Page: General Information</td>
</tr>
<tr>
<td>Manufacturer</td>
<td>Field: <strong>Manufacturer</strong></td>
<td>Field: <strong>Company Name</strong></td>
</tr>
<tr>
<td></td>
<td>Page: Product Details</td>
<td>Page: General Information</td>
</tr>
</tbody>
</table>
● If files in the captured installation depend on certain merge modules, the Download Redistributables wizard appears. Use the Wise Web Site option to download those merge modules, which should be pre-selected.

See Downloading Redistributable Files in the Windows Installer Editor Help.

● If a file that is part of a merge module is added, the Files in Merge Modules dialog box appears. It prompts you to add the merge module and, if necessary, download it.

See Adding Merge Modules Instead of Files.

● If a file that is used by a package in the Wise Software Repository is added, the Files in Repository dialog box appears and prompts you to add a version of the file that is in the repository. Because this repository check can slow the capture of large applications, you can disable it by clearing the Enable checking of files against Wise Software Repository check box in the General Settings of SetupCapture Configuration.

See Adding Files From the Wise Software Repository in the Windows Installer Editor Help.

Adding Merge Modules Instead of Files

The Files in Merge Modules dialog box appears when a file that is part of a merge module is added to the repackaged installation. Typically, it appears after you add a file to the Files page in Windows Installer Editor, or after a SetupCapture. The Files in Merge Modules dialog box lets you add the merge module that contains the file instead of adding the file.

Example:

The file olepro32.dll is part of a merge module named oleaut32.msm (Microsoft OLE 2.40). Because the file olepro32.dll is meant to function as part of a more comprehensive merge module, it is better to add the merge module instead of the individual file.

The merge module might contain other files, registry keys, and dependencies on other merge modules. If it contains dependencies, the dependent merge modules are added also. Other files and registry keys that are in the merge module are removed from the installation to avoid duplication.

To add the merge module that contains the file

Mark the check boxes of merge modules to add. To see what dependent merge modules will be added, expand the folders. Then click OK.

If the merge modules or their dependencies are not found, the Download Redistributables wizard opens with the appropriate merge modules selected. When you finish the download, the merge modules are added to the installation.

To add the file instead of the merge module

Clear the check boxes of all merge modules and click OK. Dependency merge modules are not added if you clear the parent merge module’s check box.

To hide this dialog box in the future

From Show this Dialog, select one of the following:
Capturing Applications

- **Hide; Replace files with merge modules matching version**
  Avoid seeing this dialog box in the future and always replace files with the corresponding merge modules. This turns the dialog box off for all instances in which it would normally appear.

- **Hide; Don’t automatically add merge modules**
  Avoid seeing this dialog box in the future and never replace files with the corresponding merge modules. This turns the dialog box off for all instances in which it would normally appear.

To make the dialog box appear again, start Windows Installer Editor, click the Prompts tab in Wise Options, and activate the dialog box.

See also:
*Downloading Redistributable Files* in the Windows Installer Editor Help

**Configuring the Installation as a New Feature**

(Windows Installer packages only.)

On the SetupCapture Finish page, you can save the results from this SetupCapture to a specific feature. This page appears during SetupCapture.

See *Capturing an Installation* on page 219.

**Why Configure an Installation as a New Feature?**

Suppose you have an application named Photo, which contains three separate applications, Print, Album, and Touchup. When you repackage Photo, you want the end users who will install this bundled application to be able to choose which of the three—Print, Album, and Touchup—they want to install. First, you run SetupCapture on Print’s installation. When the SetupCapture Finish page appears, click New and configure a new feature named Print. Then, do the same with Album’s and Touchup’s installations. The three applications within Photo are now features and the end user can select them during installation.

**To place installation resources in a new feature:**

On the SetupCapture Finish page, click New, then complete the Feature Details dialog box that appears. For details on the fields on this dialog box, see *Feature Table* in the Windows Installer SDK Help.

**Note**
Some options on this dialog box set the default only; the end user can change the default during installation. To prevent the end user from being able to change the defaults you set, turn off the Select Feature dialog box on the Dialogs page in Windows Installer Editor, set features to be required, or set features to be invisible.

- **Name**
  Enter the name of the feature, which is used internally by Windows Installer. The feature name is limited to 32 characters.

- **Title**
  Enter text to identify the feature. This text appears on the Select Features dialog box during installation.
- **Parent**
  This list contains all features in the installation. To change the feature’s parent, and therefore the feature tree, select from this list. This lets you change the feature tree in Installation Expert or in Setup Editor instead of editing tables.

- **Target Platform**
  Specify the platform on which this feature should be installed.

**Note**
If you add a 64-bit component to a 32-bit feature, it will never be installed. A 64-bit component will be ignored when installing on a 32-bit computer, and a 32-bit feature will not be installed on a 64-bit computer.

- **All Processors**
  The feature appears for installation on any computer, regardless of the platform.

- **32 Bit Processors**
  The feature appears for installation on 32-bit computers only.

- **64 Bit Processors**
  The feature appears for installation on 64-bit computers only.

- **x64 Only**
  The feature appears for installation on computers that support the x86 architecture (including AMD64 or EM64T).

- **Itanium Only**
  The feature appears for installation on computers that support the Itanium 64-bit processor.

- **Description**
  Enter a multi-line description of the feature. This appears if the end user selects a feature on the Select Features dialog box during installation. This text must fit in the Feature Description area of the Select Features dialog box.

- **Level**
  If you are using the Installation Types page to determine which features to install for a Typical or Complete installation, you can skip this field. If not, specify whether this feature is installed for a Typical or Complete installation. The end user chooses Typical, Complete, or Custom on the Installation Type dialog box (also called Select Installation Type). During a Custom installation, the end user can turn features on or off individually.

  Each installation has an installation level, stored in the property INSTALLLEVEL. Each feature has its own installation level value, which is set by this field. If a feature’s level is less than or equal to the installation’s INSTALLLEVEL property, then the feature is installed. By default, INSTALLLEVEL is set to 3 for a Typical installation, and to 1000 for a Complete installation.

  - **Normal**
    Set the feature’s level value to 3, which means that it gets installed by default for either Typical or Complete.

  - **Never install this feature**
    Set the feature’s level value to 0, which means that it won’t appear during installation, and won’t be installed.
- **Always install this feature**
  Set the feature’s level value to 1, which means that it gets installed by default for either Typical or Complete.

- **Custom**
  Set the feature’s installation level value yourself. Example: If you want a feature to be installed for a Complete installation, but not for a Typical installation, set a custom level value that’s greater than 3 and less than or equal to 1000. The **Custom Value** field becomes available.

- **Custom Value**
  If **Custom** is selected in **Level** above, enter the level value for the feature in this field. For details, read the description of **Level**, above.

**Note**
The end user’s action on the Installation Type dialog box determines the INSTALLLEVEL property. To see how this works, in Windows Installer Editor, go to Setup Editor > Dialogs tab and click the Installation Type Dialog in the list. Double-click the Typical/Complete/Custom radio button, and you see that the end user's choice in this radio group sets the property InstallMode. If you then double-click Next on the Installation Type dialog box and view the Events tab, you see that, based on the InstallMode value, the installation property INSTALLLEVEL is set to either 3 or 1000.

- **Display**
  Specify if and how the feature is displayed to the end user on the Select Feature dialog box during installation.
  
<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invisible</td>
<td>Do not display the feature.</td>
</tr>
<tr>
<td>Visible and Expanded</td>
<td>Display the feature and its children.</td>
</tr>
<tr>
<td>Visible and Collapsed</td>
<td>Display the feature but not its children.</td>
</tr>
</tbody>
</table>

- **Attributes**
  Specify the default for the feature in the installation. The end user can change the default.
  
<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Favor Local</td>
<td>The feature should be installed on the destination computer.</td>
</tr>
<tr>
<td>Favor Source</td>
<td>The feature should be run from the source CD or network directory. This means the feature is available to the application, but is not installed on the local hard drive. When the feature is invoked, your application must call Windows Installer functions (example: MsiGetComponentPath) to locate and read the necessary files from the installation source, which might be a CD or shared network directory. A typical use of this option would be to specify a clip art library to run from the source. Then you must code your application to try to read from the installation source when the end user tries to use the clip art library.</td>
</tr>
<tr>
<td>Favor Parent</td>
<td>The feature should use the same attribute setting as its parent feature. If you select this option, you must also set the installation to generate uncompressed external files that can run from the source. See the description of <strong>Media Type</strong> in Adding a Media Item in the Windows Installer Editor Help.</td>
</tr>
</tbody>
</table>
Capturing Applications

- **Advertising**
  Specify the default setting for how this feature supports advertising. If a feature is advertised, it is not installed, but it appears to be installed to the end user. Example: the end user might see shortcuts or menu options for an advertised feature, or the system might have certain entry points to the feature, such as a registered file extension, that can start installation-on-demand of an advertised feature.
  - **None**
    By default, the feature is set to be installed, not advertised.
  - **Favor Advertising**
    By default, the feature is set to be advertised.
  - **Disallow Advertising**
    The feature is set to be installed, not advertised, and the end user cannot change the default.

  **Note**
  Advertising is only supported on certain platforms. To suppress advertising on unsupported platforms, mark **Disable advertising if not supported by OS** below. See Platform Support of Advertisement in the Windows Installer SDK Help.

- **Directory**
  To let end users select the directory for this feature, select a directory from this list. When the end user selects this feature on the Select Feature dialog box during installation, the Browse button becomes enabled so that they can select a new directory. Child features of this feature inherit the new directory selected by the end user. If you leave the directory set to `<none>`, then the files for this feature are installed in the directory structure specified on the Files page in Windows Installer Editor.

  To ensure that two features always get installed to the same directory, select the same option in **Directory** for both features.

  If you let end users select directories for individual features, you must code your application in such a way that it can locate the features wherever they might be placed by the end user. To do this, you can call Windows Installer functions, such as MsiGetComponentPath.

  **Note**
  Only the files that are in the directory you select or in its child directories will be installed in the new directory that the end user selects. Example: Suppose FeatureA installs File1 in the Sample\FeatureA\ directory and File2 in the Windows directory. During installation, the end user specifies Sample\A\ for the new directory. Only File1, which was originally in the FeatureA\ directory, is actually installed in the A\ directory. File2 is still installed in the Windows directory.

- **Required Feature**
  Mark this if the feature is required for installation. During installation, end users do not have the option to deactivate installation of a required feature. If you select **Never install this feature in Level** (above), it overrides this option.

- **Disable advertising if not supported by OS**
  Mark this to ignore any choices you’ve made in **Advertising** if the operating system on the destination computer does not support advertising. See Platform Support of Advertisement in the Windows Installer SDK Help.
Using SetupCapture With Virtual Capture

Not available in Standard Edition.

Use Virtual Capture to capture on a non-clean machine while achieving clean machine results. This eliminates the need to reimage your computer between each SetupCapture.

You also can eliminate the need to reimage a clean machine by capturing applications in a virtual software layer. While Virtual Capture requires that you create a Virtual OS, the only requirement for capturing applications into a virtual software layer is the installation of the Software Virtualization Agent, which is installed with Wise Package Studio.

See Capturing an Installation in a Virtual Software Layer on page 219.

Using Virtual Capture is a two-step process:

- Run the Virtual OS Creation utility to create a Virtual OS.
  See Creating a Virtual OS on page 239.
- During SetupCapture, select Virtual Capture as the capture method, and specify the Virtual OS.
  See Capturing an Installation on page 219.

The same process is used to import any non-Windows Installer package into the Software Manager database. To do this, select the Universal Import option in the Import Wizard in Software Manager and specify a Virtual OS.

See Performing a Universal Import Without Converting or Repackaging in the Software Manager Help.

Note
A security setting in Windows Vista or later operating system prevents Virtual Capture from working. When you try to use Virtual Capture, a prompt appears and provides the option to disable the security restriction. If you choose to disable the security restriction, the following registry setting is set:

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Windows>LoadAppInit_DLLs=1

The initial default for this setting is 0. If you disable this restriction, your computer’s vulnerability to malicious attack is increased. However, if you run SetupCapture in a testing environment, the increased vulnerability might not be a critical issue.

Options for Creating and Using the Virtual OS

If you plan to perform captures on the current clean machine, create the Virtual OS on the current computer. This creates a Virtual OS directory and registry structure where the application will be installed during SetupCapture.

If you plan to perform captures on a non-clean machine, create the Virtual OS on a clean machine and save a Virtual OS file (.WOS) in a shared location. When you run SetupCapture with Virtual Capture on the non-clean machine, select the Virtual OS file. It is expanded onto the capture computer as a Virtual OS directory and registry structure where the application will be installed during SetupCapture. This process takes several minutes and requires a substantial amount of free disk space.
How the Virtual OS Works

A virtual OS consists of a simulated directory structure and a simulated area of the registry. The complete clean machine is duplicated in this directory and registry structure, which takes a substantial amount of free disk space. When SetupCapture runs an installation, all files and registry entries are redirected to the simulated directory structure and registry, preventing changes from actually happening on your computer.

During each subsequent session of SetupCapture, the Virtual OS is cleaned. This means that installed remnants from the last capture are removed from the Virtual OS directory and registry structure. This lets you perform multiple captures on the same computer without reimaging between each and without having previous capture results interfere with the current capture.

See also:

Guidelines for Virtual Capture on page 238

Guidelines for Virtual Capture

➤ Not available in Standard Edition.

- The Virtual OS Creation utility should only be run on a clean machine.
  See Setting Up a Clean Machine on page 218.
- Virtual Capture can only be used on Windows 2000 and later.
- The computer on which you use Virtual Capture or perform a Universal Import must have the same operating system as the computer on which the Virtual OS was created. Example: If you ran the Virtual OS Creation utility on a computer with Windows 2000, service pack 2, MDAC 2.5, and Internet Explorer 5, then the computer where you run SetupCapture with Virtual Capture must also have Windows 2000, service pack 2, MDAC 2.5, and Internet Explorer 5.
- The contents of the computer on which you create the Virtual OS cannot exceed 3 GB of used space.
- Execute the original application from a location other than the drive that contains the Virtual OS file (which is always the drive with Windows). Example: If your C drive contains Windows and your WiseImg directories, then execute the application from a network location, a second drive, or a CD drive.
- When you create a Virtual OS, run Virtual Capture, or perform a Universal Import, you need enough free disk space to completely duplicate the computer’s contents, plus extra space for virtual memory and for installing applications that you capture.
- The virtual directory structure must be on the same drive letter (partition) as the Windows directory.
- While using Virtual Capture, it is very important to start installations from the Execute or Next button on the Execute Installation page of SetupCapture. While performing a Universal Import in Software Manager, you must start installations from the Next button on the Package Details page. If you start them outside the Execute Installation or Package Details page, SetupCapture or Universal Import is unaware of the changes that occur.
- The option to capture into a virtual software layer is not available with the Virtual Capture method.
Capturing Applications

Creating a Virtual OS

Not available in Standard Edition.

Before you use SetupCapture with Virtual Capture, use the Virtual OS Creation utility to create a Virtual OS.

See Guidelines for Virtual Capture on page 238.

The Virtual OS Creation utility makes a copy of the entire computer, except Recycler and files that have system attributes, such as pagefile.sys.

To create a virtual OS

1. Copy the VirtualOS.exe file to a clean machine. It is located in the Wise Package Studio\Workbench directory.

2. On the clean machine, open VirtualOS.exe.

   The Virtual OS Location page appears.

3. In Name of Virtual OS, enter a descriptive name to identify this Virtual OS. (Example: Clean Windows XP.) This name will be displayed in SetupCapture and the Universal Import option in the Import Wizard.

4. Specify where to locate the Virtual OS:

   - Create a Virtual OS on this PC
     Select this option if you plan to perform captures on the current clean machine. This would be the case if you are working in a clean build environment.

   - Create a Virtual OS file and copy to a shared location
     Select this option if the current computer is clean, but you plan to perform captures on another, non-clean machine. Also select this option if you plan to use the Universal Import feature in Software Manager. Click Browse to specify a location for the Virtual OS file. The file will be extremely large, so select a location with plenty of disk space.

5. Click Next.

   Creation of the Virtual OS begins. This can take several minutes to complete. When it is finished, “Current Status is Scan Completed” appears on the page.

6. Click Finish.

   - If you created the Virtual OS on this computer, your hard drive contains a new directory, which contains a copy of all files on this computer. Also, a new registry branch contains a copy of the entire registry.

   - If you created a Virtual OS file, then a .WOS file is created in the location you specified. This file is large because it contains a copy of the registry and of every file on the computer.

You can now perform a SetupCapture using Virtual Capture.

See also:

Using SetupCapture With Virtual Capture on page 237
Creating a Virtual OS on page 239
Using SetupCapture to Capture First Use Settings

You can capture the changes made by an .MSI-based application the first time it is started on a computer after being installed. (Example: preferences, serial numbers that are required on first use, and so on.) The computer must have a freshly installed application, which has not yet been started.

A transform is created that contains the first use settings, which can be applied to the base .MSI that originally installed the application to simulate the changes made during the first run. Example: Suppose your company has a site-wide license for an application that always asks for a serial number upon first run. Rather than distribute the serial number to each user, use SetupCapture to create a transform that simulates the entering of the serial number.

To capture first use settings into a transform
1. Exit all other applications so changes they make are not captured.
2. On the Tools tab, double-click SetupCapture.
3. On the SetupCapture Type page, specify the following, then click Next:
   - **First Use Settings**
   - **Target .MST File**
     Specify the file name and location of the transform file that will be created.
   - **Base .MSI File**
     Specify the .MSI that installed the application.

The Welcome page appears.

4. On the Welcome page:
   - To select a different configuration file, click Change. This session of SetupCapture will use the settings in the specified configuration file. See Selecting the Configuration File on page 205.
   - To change the configuration settings for the current SetupCapture, click Settings. You can save those changes to the configuration file (optional). See:
     - Setting General Settings on page 205
     - Setting Directories to Watch on page 208
     - Setting File and Folder Exclusions on page 211
     - Setting Registry Exclusions on page 214
     - Setting INI File Exclusions on page 215

      If the Change button or the Settings button is unavailable, you might not have permission to change settings. See Setting SetupCapture Configuration Security on page 44.

5. Click Next on the Welcome page.
6. On the Capture Methodology page, select a method for capturing installations and click Next. Virtual Capture is not available on this page because it is not applicable to capturing first use settings.
Capturing Applications

See *Selecting the Capture Methodology* on page 224.

If you selected the snapshot method, the Initial Scan page might appear. Specify whether to rerun the initial scan or use the initial scan that was created previously.

See *Using a Previous Scan* on page 227.

The Begin Application Capture page appears.

7. Click Next.

What happens next depends on the capture methodology you chose:

- **SmartMonitor**: Monitoring begins.
- **Snapshot**: Your computer is scanned, unless you are using the previous initial scan. Do not work on the computer during the scan.

The Execute the Application page appears.

8. On the Execute the Application page, specify an application. After you make your selections, click the Execute button to run the application. To run another application, select another shortcut or executable and click Execute again.

- **Application Shortcuts**
  This lists any shortcuts that were installed with the application. If this list is empty, make sure that the application is already installed.

- **Application Executables**
  Use this option to browse to each application executable to run.
  
  - **.EXE Name**
    Specify the full path of the installation executable.
  
  - **Command Line**
    Enter any command lines to apply to the executable.

- **Capture the application in a virtual software layer**
  If you mark this, all changes made to the computer when you set first use settings are put into a virtual software layer. You can then use Symantec SVS applet to delete or deactivate the layer and restore the computer to its original state.

  See *Capturing an Installation in a Virtual Software Layer* on page 219.

When you run an application, make sure you exercise any features that set initial preferences, such as turning tips on or off, specifying interface styles, and fulfilling any first time only prompts, such as serial numbers.

9. Click Next on the Execute Application page.

The End of the Application Capture page appears.

10. Click Next on the End of the Application Capture page.

    (Standard Edition.) The wizard closes—no additional pages appear. The transform is saved in the directory you specified at the beginning of this procedure. Skip the next steps that describe the Inclusions and Exclusions pages.

    (Professional or Enterprise Editions.) The First Use Settings Inclusions page appears.

    The First Use Settings Inclusions page displays the items that will be added to the transform. These items represent changes that were detected during scanning or monitoring.
11. On the First Use Settings Inclusions page, to remove an item from the transform, select it in the list and click Exclude. The Exclude Globally button causes it to be excluded from future captures. To display another type of item, select the type from Inclusion Type.

See Editing Setup Capture Inclusions on page 228.

12. Click Next on the First Use Settings Inclusions page.

The First Use Settings Exclusions page appears. It displays all the changed items that are excluded from the transform based on the exclusion list.

13. On the First Use Settings Exclusions page, to add an item back into the transform, select it in the list and click Include. To display another type of item, select the type from Inclusion Type.

See Editing Setup Capture Exclusions on page 229.

14. Click Finish.

- If files in the captured installation depend on certain merge modules, the Download Redistributables wizard appears. Use the Wise Web Site option to download those merge modules, which should be pre-selected.

  See Downloading Redistributable Files in the Windows Installer Editor Help.

- If a file that is part of a merge module is added, the Files in Merge Modules dialog box appears. It prompts you to add the merge module and, if necessary, download it.

  See Adding Merge Modules Instead of Files on page 232.

The transform is saved in the directory you specified at the beginning of this procedure.

**SOE Snapshot**

➤ Not available in Standard Edition.

SOE Snapshot lets you capture a computer’s standard operating environment (SOE). An SOE consists of the operating system, basic system software, and the applications that are installed for every user or every user in a department. SOE Snapshot stores this snapshot in the form of an .SOE file, which you can import into the Software Manager database to represent a baseline computer in your organization.

Example:

Suppose you capture the SOE and import it into the Software Manager database. Then you capture an application, import it, and perform a conflict analysis. You might find conflicts between the application and the SOE. ConflictManager treats the SOE as another package against which you can find conflicts.

See:

Guidelines for Capturing the Standard Operating Environment on page 243

SOE Snapshot Configuration Settings on page 243

Capturing the Standard Operating Environment on page 244
Guidelines for Capturing the Standard Operating Environment

- **Not available in Standard Edition.**

  - Capture on a baseline machine that has the operating system and the applications that are installed for every user or every user in a department. If you only want to capture the operating system, capture on a clean machine.  
    See *Setting Up a Clean Machine* on page 218.

  - Before running SOE Snapshot, exit all other applications.

  - Before starting the capture, consider increasing the size of the virtual memory page file. Capturing a standard operating environment is a CPU-intensive process that uses a lot of memory. See your operating system documentation for instructions.

  - The SOE Snapshot process might take a long time, depending on the amount of software on the computer you are capturing.

  - If you open an .SOE file in Windows Installer Editor, some operations, such as displaying the Registry page, might take longer than usual because of the large amount of data.

  - Although you can edit an .SOE file in Windows Installer Editor, you cannot compile it into an executable installation. The sole purpose of an .SOE file is to let you import a snapshot of a baseline machine into the Software Manager database.

See also:
- *SOE Snapshot Configuration Settings* on page 243
- *Capturing the Standard Operating Environment* on page 244

SOE Snapshot Configuration Settings

- **Not available in Standard Edition.**

SOE Snapshot uses exclusion settings from a configuration file created by SetupCapture Configuration.

You can create a configuration file using SetupCapture Configuration prior to running SOE Snapshot or you can specify settings when you run SOE Snapshot. You also can select a different configuration file when you run SOE Snapshot.

See the following topics:

  - *SetupCapture Configuration* on page 202
  - *Setting File and Folder Exclusions* on page 211
  - *Setting Registry Exclusions* on page 214
  - *Setting INI File Exclusions* on page 215

By default, all directories on the computer are scanned, which produces the most complete and accurate representation of the standard operating environment. SOE Snapshot automatically ignores certain system files.

See *Files and Registry Entries Ignored During Captures* on page 247.
Capturing the Standard Operating Environment

➤ Not available in Standard Edition.

Use SOE Snapshot to capture a computer’s standard operating environment (SOE). Before you run SOE Snapshot, see Guidelines for Capturing the Standard Operating Environment on page 243.

Note
Before you run SOE Snapshot, you can run SetupCapture Configuration to create a configuration file to be used by SOE Snapshot.

See SOE Snapshot Configuration Settings on page 243.

To capture the standard operating environment

1. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with SOE Snapshot. This tool might skip pages or populate fields based on command-line options defined in Process Templates Setup.
   - On the Tools tab, double-click SOE Snapshot. The Welcome page appears. The Configuration File Location shows the configuration file that will be used for this SOE Snapshot.

2. On the Welcome page:
   - To change the configuration settings for this capture without changing the current configuration file, select the Do not change current configuration file option. This option is enabled only if the user has permission to use the SetupCapture Configuration tool. If the Settings button is disabled, selecting this option enables it. The user can then modify the settings for the current capture.
   - To select a different configuration file, click Change. This session of SOE Snapshot will use the settings in the specified configuration file. See Selecting the Configuration File on page 205.
   - To change the configuration settings for the current SOE Snapshot, click Settings. By default, anything you change on all tabs of this dialog box is saved to the current configuration file and affects future captures. If you don’t want to change the configuration file, mark the check box Do not change current configuration file. See:
     Setting File and Folder Exclusions on page 211
     Setting Registry Exclusions on page 214
     Setting INI File Exclusions on page 215
   - If the Change button or the Settings button is unavailable, you might not have permission to change settings. See Setting SetupCapture Configuration Security on page 44.

3. Click Next to display the SOE Scan page.

4. Click Next to begin scanning the SOE. This can take several minutes.
   - When the scan is complete, the SOE Snapshot Inclusions page appears. It displays the files, registry keys, .INI files, and shortcuts that were captured during the SOE scan. These items will be included in the .SOE file.
5. Edit the inclusions as needed:
   - From **Inclusion Type**, select the type of inclusion to display.
   - To exclude an item from the SOE Snapshot, select the item and click Exclude.
   - To add an item to the exclusion list for the current configuration file, select the item and click Exclude Globally. If the Exclude Globally button is unavailable, see *Setting SetupCapture Configuration Security* on page 44.

6. When you finish editing inclusions, click Next.

   The SOE Snapshot Exclusions page appears. It displays the files, registry keys, .INI files, and shortcuts that were excluded from the SOE Snapshot, based on the exclusion list that is defined in SetupCapture Configuration settings.

   See *SetupCapture Configuration* on page 202.

7. Edit the exclusions as needed:
   - From **Exclusion Type**, select the type of exclusion to display.
   - To include an item in the SOE Snapshot, select the item and click Include.

8. When you finish editing exclusions, click Next.

   The Finish page appears.

9. In **File Name**, specify the name and location of the .SOE file and then click Finish.

   An .SOE file containing the SOE snapshot is created in the location you specified. You can use Software Manager to import this file into the Software Manager database. Then, in ConflictManager, you can check for conflicts between other applications and your SOE.

   **Note**
   You might see a message telling you that a service in the capture requires a password to function correctly. This happens when a service on the computer is installed under an account other than the system account. Because you won’t be installing the result of an SOE Snapshot onto another computer, it is not necessary to enter a password.

---

### Capturing With Wise Web Capture

> *Not available in Standard Edition.*

Wise Web Capture, which you run from a browser, lets you capture installations on a clean machine without installing any additional software. It also lets you capture on a computer that is running a non-supported operating system.

**Example:**

A repackager manages a large group of computers on which captures are performed. Before a capture, each computer is re-imaged. The repackager cannot change the image in any way. By using Wise Web Capture, the repackager can quickly re-image the computer and perform the capture without pre-installing capture software.

Wise Web Capture records all the changes performed by an installation and saves that information to a new Windows Installer package. It does this by scanning the computer before and after the installation and recording the differences between the two scans. (This is the same as the snapshot method that SetupCapture uses.) Wise Web Capture uses a predefined exclusion list to determine which items to ignore during the capture.
The file that results from the capture is an encrypted .MSI, with the extension .MSI_. You cannot open or install this encrypted file, but you can decrypt it on a computer that has Wise Package Studio installed.

**Requirements**
- Version 2.0 of the .NET Framework must be installed on the server.
- ASP and ASP.NET must be installed and enabled on the server.
- The Wise Web Capture Web application must be installed on a Microsoft Internet Information Services (IIS) Web server on your network.
- When capturing on a computer running Windows Vista or later operating system, you must uncheck Enable Protected Mode on the Security tab of the Internet Options.

Wise Package Studio has additional requirements.

See *System Requirements* in the *Wise Package Studio Getting Started Guide*.

**To capture an installation**
1. Close all other applications.
2. In your browser, enter the URL for Wise Web Capture, which is:
   - http://machine name/Wise_Web_Capture
     where *machine name* is the computer on which the Wise Web Capture Web application is installed. Example: http://localhost/Wise_Web_Capture
3. On the Log On page, enter a valid Workbench user name and password and click Submit.
4. On the Specify Output page:
   - In **Target Installation**, specify the full path of the file in which to store the capture results. Do not include a file extension; the extension .MSI_ is appended.
   - Click Next to begin the initial scan.
5. After the initial scan, leave the browser window open and run the installation to be captured.
6. When the installation finishes, return to the browser window and click Next to begin the final scan.
7. The captured installation is saved to the file you specified.

**To decrypt the captured file**
The computer on which you decrypt the file must have Wise Package Studio installed.

You can decrypt the captured file in either of two ways:
- In Windows Explorer, double-click the .MSI_ file.
- In Wise Package Studio Workbench:
  - Run the Web Capture Conversion tool from the Tools tab or Projects tab.
  - On the Select File to Convert dialog box, specify the .MSI_ file and click OK.

The file is decrypted and the extension is renamed to .MSI.
Files and Registry Entries Ignored During Captures

A computer typically contains system files and registry entries that should not be included in the capture of an application or standard operating environment. Some of these files are Wise product-specific and others are computer-specific.

What is Excluded from SetupCapture, SOE Snapshot, and the Test Expert Machine Capture?

- Items that are listed in the WisePSSC.ini file, which is located in the Windows or Winnt directory. Alternatively, the file might be named repackage.ini. This extensive exclusion list contains:
  - Predefined exclusions, including all Wise files and registry entries as well as files and registry entries that would be dangerous to install on another computer.
  - Files or registry entries that you added to the exclusion list in SetupCapture or SetupCapture Configuration.
  - Files or registry entries that you added to the exclusion list on the Machine Capture Settings dialog box in Test Expert.
- Files and registry keys that are hard-coded to be ignored.
  See Files that are hard-coded to be ignored on page 247.
  See Registry keys that are hard-coded to be ignored on page 248

Files that are hard-coded to be ignored

An asterisk in any of the following paths means that any file in the directory is excluded.

- User Profile\ntuser.dat
- User Profile\Ntuser.dat.log
- User Profile\Local Settings\Application Data\Microsoft\Windows\UsrClass.dat
- User Profile\Local Settings\Application Data\Microsoft\Windows\UsrClass.dat.log
- User Profile\Local Settings\Application Data\IconCache.db
- User Profile\Recent\*
- User Profile\Local Settings\Temporary Internet Files\*
- All Users Profile\Application Data\Microsoft\Network\Downloader\*

- BootLog.txt
- PageFile.Sys
- 386spart.par

- Windows\Win386.swp
- Windows\User.dat
- Windows\User.dao
- Windows\System.dat
- Windows\System.dao
- Windows\Bootstat.dat
- Windows\Setupapi.log
- Windows\Schdlgu.txt
- Windows\Fntcache.dat
- Windows\Windows Update.log
- Windows\Tfcache\*
- Windows\ShellIconCache\*
Capturing Applications

Windows\Installer\*
Windows\Security\*
Windows\Debug\*
Windows\CSC\*
Windows\Prefetch\*
Windows\Applog\*
Windows\LastGood\*

Windows\System32\Config\*
Windows\System32\Wbem\Logs\*
Windows\System32\Wbem\Repository\*
Windows\System32\Catroot\*
Windows\System32\Catroot2\*

Program Files\Windows Update\*
Recycler\*
Recycled\*
System Volume Information\*

The following items are hard-coded to be ignored by SetupCapture and the Text Expert Machine Capture but not SOE Snapshot. However, they might be in the predefined exclusion list in WisePSSC.ini. Therefore, if you do want these items to be captured by SOE Snapshot, remove them from the predefined exclusion list.

Windows\System32\Config.log
Windows\System32\Shlwapi.dll
Windows\System32\Shdocvw.dll
Windows\System32\Shfolder.dll
Windows\System32\Dllcache\*

**Registry keys that are hard-coded to be ignored**

SetupCapture, SOE Snapshot, and the Test Expert Machine Capture ignore the following registry hive:

HKLM\SOFTWARE\Wow6432Node\ and all its subkeys

Changes that were made to the Windows Vista or later operating system caused 32-bit processes and applications to see a recurring Wow6432Node registry hive. This situation caused the SetupCapture snapshot scan to hang and crash. When this hive is ignored, these problems are prevented.
Chapter 9
Package Distribution

This chapter includes the following topics:
- About Package Distribution on page 249
- Distribution Methods on page 250
- Moving a Package into Microsoft Active Directory on page 250
- Copying a Package to the Share Point Directory on page 252
- Copying a Package to a Network Directory on page 254
- Copying a Compiled Installation to an FTP Server on page 256
- Performing an Administrative Installation of a Windows Installer Package on page 258

About Package Distribution

Use Package Distribution to deploy or share a package by:
- Copying a package to the share point directory or an FTP server.
- Copying a compiled installation or a project and its associated files to a network directory.
- Creating a Windows Installer administrative installation.
- Preparing a package's .MSI file for distribution to end users using Microsoft® Active Directory.

You can select these types of packages to distribute:
- Windows Installer packages: .WSI, .MSI
- Merge modules: .WSM, .MSM
- Windows Installer patches: .MSP
- Transforms: .MST
- WiseScript packages: .WSE (Professional Edition only.)
- Virtual software packages: .WVP, .VSA (Professional Edition only.)

Note
You cannot distribute device driver (.INF) files or Group Policy Objects (.POL) that are in the Software Manager database. However, you can create a Group Policy Object directly, using the Active Directory option in Package Distribution.

You can run Package Distribution from Workbench or from Software Manager. What’s the difference? Typically, when you run Package Distribution from Workbench, you distribute a project file or the compiled installable file from its source directory, such as the Workbench project directory. When you distribute from Software Manager, you...
distribute installable files only (examples: .MSI or .EXE files) from the Software Manager database. Therefore, when you run Package Distribution from Software Manager, the options to distribute to the share point directory and to distribute a project file are not available.

Distribution Methods

When you run Package Distribution, you select a distribution method. The following criteria determine whether a particular distribution method is available:

- Whether you run the tool from Workbench or Software Manager.
- The type of package you select to distribute.
- The edition of Wise Package Studio you have installed.
- Whether or not you are distributing a preflight package (generated from the Preflight Instrumentation tool).
  If you are, then the options to distribute to the share point, the network (project file), and Administrative Installation are unavailable.

The distribution methods are described in the following topics:

- **Moving a Package into Microsoft Active Directory** on page 250
- **Copying a Package to the Share Point Directory** on page 252
  (Not available in Standard Edition.)
- **Copying a Package to a Network Directory** on page 254
- **Copying a Compiled Installation to an FTP Server** on page 256
- **Performing an Administrative Installation of a Windows Installer Package** on page 258

Moving a Package into Microsoft Active Directory

When you are ready to distribute a Windows Installer package to end users, use Package Distribution to move a Group Policy Object for the package into Microsoft® Active Directory. Once the Group Policy Object is in Active Directory, you must link the object to the organizational unit that the package should be applied to.

The following criteria determine whether this option is available.

<table>
<thead>
<tr>
<th>You specify this type of file:</th>
<th>.MSI, .WSI</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If you specify a .WSI, it is compiled and the compiled installation is distributed.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>You run Package Distribution from:</th>
<th>Workbench</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Software Manager</td>
</tr>
</tbody>
</table>
To move a package into Microsoft Active Directory

1. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with Package Distribution. The package associated with the current project will be distributed. This tool might skip dialog boxes or pre-fill fields based on command-line options defined in Process Templates Setup.
   - On the Tools tab, double-click Package Distribution. On the Specify File to Distribute page that appears, specify a file and click Next. Use UNC or mapped drive notation.
   - In Software Manager, in the Applications/Packages pane, select a package. Then select Packages menu > Distribute. The Distribution Method page appears.

2. If you specified a .WSI that contains multiple releases, a drop-down list appears. Select a release.

   If this option is not available, you might not have Active Directory set up on your computer.
4. Click Next. If necessary, the installation file is saved and compiled.

   The Microsoft Active Directory dialog box appears.
5. Complete the dialog box:
   - **MSI Pathname**
     This non-editable field contains the file name of the package you chose to distribute. This must be specified using UNC or mapped drive notation, to ensure that it is accessible to all users in the Group Policy Object; see requirements listed above.
   - **User Name**
     Enter an account name that has privileges to create a Group Policy Object on the Active Directory server for a particular domain. Example: DOMAIN1\Administrator
   - **Password**
     Enter the password for the user account specified above.
   - **Computer**
     Enter the computer name of the Active Directory server.
   - **Domain**
     Enter the domain of the computer specified above.

Other requirements:
- Active Directory must be set up on the computer that is running Wise Package Studio. You must have privileges to make changes in the Active Directory management console of an Active Directory server.
- If distributing from the Projects tab or Software Manager, the share point directory must have been specified during installation using UNC or mapped drive notation.
Package Distribution

- **GPO Name**
  Enter a unique name for the Group Policy Object to be created. Adhere to Microsoft Group Policy Object naming conventions.

- **Settings**
  Set the package to be deployed per machine or per user. This determines where an application's configuration information is stored.

- **Deployment Method**
  If you selected **Per User** above, you can set the package to be assigned or published. If an application is assigned to a user, it appears installed, including shortcuts, extensions, icons, and registry entries. Publishing can only occur from a Windows Server and does not populate the interface. Publishing only registers extensions and MIME types. In both cases, installation-on-demand takes place when the user invokes an entry point to the application.

  If you selected **Per Machine** in the **Settings** field, this field is set to **Assign** and is unavailable, because of Active Directory restrictions.

6. Click Finish.

A new Group Policy Object is created that contains the installation. It is added to the list of all Group Policy Objects stored in the domain but is not assigned to any organizational unit—to add it, use the Active Directory management console.

### Copying a Package to the Share Point Directory

You can use Package Distribution to copy a package or merge module to the share point directory, for later importing into the Software Manager database. Distributing to the share point directory also copies all of the package’s source files to the share point and updates the source paths for those files, which means that after resolving conflicts you can recompile the fixed package from its location on the share point.

Distributing to the share point directory does not actually import the package into the Software Manager database, but copies it to a directory in a format that can be imported into the Software Manager database. You use Software Manager to import it from the share point.

When you distribute a .VSA file to the share point directory and import it into the Software Manager database, it is converted to a .WVP file.

After you copy a package or merge module to the share point directory and import it into the Software Manager database, you can use Software Manager to manage the package and use ConflictManager to check the package for conflicts.

The following criteria determine whether this option is available.

<table>
<thead>
<tr>
<th>You specify this type of file:</th>
<th>.WSI, .MSI, .WSM, .MSM, .WSE, .WVP, .VSA</th>
</tr>
</thead>
<tbody>
<tr>
<td>You run Package Distribution from:</td>
<td>Workbench only.</td>
</tr>
<tr>
<td>Other requirements:</td>
<td>• Not available in Standard Edition.</td>
</tr>
<tr>
<td></td>
<td>• Not available if distributing a preflight package (generated with Preflight Instrumentation tool.)</td>
</tr>
</tbody>
</table>
Note
If you add files to a package that has been distributed to the share point directory, you are prompted to add the new files to the share point. If you do so, the .QUE file for that package is reset and you must re-import the package in Software Manager.

To copy a package to the share point directory
1. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with Package Distribution. The package associated with the current project will be distributed. This tool might skip dialog boxes or pre-fill fields based on command-line options defined in Process Templates Setup.
   - On the Tools tab, double-click Package Distribution. On the Specify File to Distribute page that appears, specify a file and click Next.

   The Distribution Method page appears.

2. If you specified a .WSI that contains multiple releases, a drop-down list appears. Select a release.

3. Mark Distribute to share point directory.

   If this option is not available, you might not have met the criteria listed at the beginning of this topic.

4. Click Next. If necessary, the installation file is saved and compiled.

   The Distribute to share point directory dialog box appears.

5. Share Point displays the share point directory to which the package will be copied. It defaults to the share point directory that is specified in Workbench Preferences.

6. The Copy Installation File To check box enables Pathname. (This option does not appear for virtual software packages.) During distribution, a copy of the package’s installation file is placed in the directory you specify here, and its source paths are updated to reflect the location of the source files in the share point directory.

   - If you run Package Distribution from the Projects tab, Copy Installation File To is cleared, because the installation file is already in the share point directory and does not need to be copied.

   - If you run Package Distribution from the Tools tab, specify a directory in Pathname:

   Warning
   If an installation file is in the share point directory, Copy Installation File To is cleared by default. If you mark it, then two copies of the installation file will be in the share point directory, which can lead to confusion about which file to edit.

   - In most cases, if the installation file is not already in the share point directory, it is best to specify the Scripts\application\package subdirectory of the share point directory, which is the default.

   - If you are distributing an .MSI or .WSI that is set to compile with external files, specify either a subdirectory of Scripts or some other directory. This prevents overwriting any external files that might also be used by another package. To verify how the package is set to compile, open the package in
Package Distribution

7. **Application Name** and **Package Name** are pre-filled except for virtual software packages and when distributing a package from WiseScript Package Editor. These names will be assigned to the package when it is imported into the Software Manager database.

8. Click Finish.

The package or merge module is copied to the share point directory. The next time someone in your organization opens Software Manager, **Queued for Import** in the Database pane will indicate that another package is waiting to be imported. If you are using the Auto Import Service, the package is imported automatically.

See *Importing From the Share Point Directory* in the Software Manager Help.

### Copying a Package to a Network Directory

When you are ready to deploy a package to end users, share it with a coworker, or copy it to another computer, you can use Package Distribution to copy a compiled installation or a project and its associated files to a network directory.

Copying a project and its source files to a network directory lets you share the project with coworkers without breaking the paths to source files. The source file paths are changed to the new paths during this process. These paths are not relative; however, you can use relative paths in a project file:

- Open a Windows Installer project in Windows Installer Editor and select Tools menu > Convert Source Paths.
- Open a WiseScript project in WiseScript Package Editor and select Edit menu > Source Directories.

The following criteria determine whether this option is available.

<table>
<thead>
<tr>
<th>When you distribute the compiled package, you specify this type of file:</th>
<th>.MSI, .MSM, .MSP, .MST, .WSI. If you specify a .WSI, it is compiled and the compiled installation is distributed. If an .EXE is associated with the installation, the .EXE is distributed.</th>
</tr>
</thead>
<tbody>
<tr>
<td>.WSE. It is compiled and the compiled installation is distributed. (Not available in Standard Edition.)</td>
<td>.WVP, .VSA. If you specify a .WVP, it is compiled and its compiled .VSA file is distributed. (Not available in Standard Edition.)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>When you distribute the project file, you specify this type of file:</th>
<th>.WSI</th>
</tr>
</thead>
<tbody>
<tr>
<td>.WSM</td>
<td>.MSI (An .MSI can be a project file if it contains source paths.)</td>
</tr>
<tr>
<td>.WSE (Not available in the Standard Edition.)</td>
<td>.WVP, .VSA. If you specify a .VSA, its project file (.WVP) is distributed. (Not available in Standard Edition.)</td>
</tr>
</tbody>
</table>
To copy a package to a network directory

1. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with Package Distribution. The package associated with the current project will be distributed. This tool might skip dialog boxes or pre-fill fields based on command-line options defined in Process Templates Setup.
   - On the Tools tab, double-click Package Distribution. On the Specify File to Distribute page that appears, specify a file and click Next.
   - In Software Manager, in the Applications/Packages pane, select a package. Then select Packages menu > Distribute.
     The Distribution Method page appears.

2. If you specified a .WSI that contains multiple releases, a drop-down list appears. Select a release.
   - Mark Network. If this option is not available, you might not have met the criteria listed at the beginning of this topic.

3. If you are running Package Distribution from Workbench, select one of the following options; they appear below the Network option. If you are running Package Distribution from Software Manager, these options do not appear and the compiled package is distributed.
   - **Compiled package**
     Distributes the compiled installation.
   - **Project file and source files**
     Distributes the project file and its source files.

4. Click Next. If necessary, the installation file is saved and compiled.
   - The Network Directory dialog box appears.

5. Complete the dialog box:
   - **Network Directory**
     Specify the directory to copy the package to.
**Destination File Name**
(Optional.) This appears if you specified a Windows Installer package. Enter an alternate name for the file that is saved to the network directory. Do not include a file extension.

6. Click Finish.

### Copying a Compiled Installation to an FTP Server

When you are ready to deploy a package to end users, you can use Package Distribution to copy the package installation to an FTP server.

Package Distribution uses the FTP protocol to transfer the files to the server location you specify. End users can download the files from the FTP server using an FTP client. If the server is also configured to run a Web server, end users can download the files through their Web browser (HTTP protocol) as well. Package Distribution does not support passive FTP, which is required by some firewall and gateway configurations, and it cannot FTP through a proxy server.

The following criteria determine whether this option is available.

<table>
<thead>
<tr>
<th>You specify this type of file:</th>
<th>.MSI, .MSM, .MSP, .MST, .WSI, .WSM. If you specify a project file, it is compiled and the compiled installation is distributed.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>.WVP, .VSA. If you specify a .WVP, it is compiled and its compiled .VSA file is distributed. (Not available in Standard Edition.)</td>
</tr>
<tr>
<td></td>
<td>.WSE. It is compiled and the compiled installation is distributed. (Professional Edition only.)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>You run Package Distribution from:</th>
<th>Workbench</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Software Manager</td>
</tr>
</tbody>
</table>

### To copy an installation to an FTP server

1. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with Package Distribution. The package associated with the current project will be distributed. This tool might skip dialog boxes or pre-fill fields based on command-line options defined in Process Templates Setup.
   - On the Tools tab, double-click Package Distribution. On the Specify File to Distribute page that appears, specify a file and click Next.
   - In Software Manager, in the Applications/Packages pane, select a package. Then select Packages menu > Distribute.
     The Distribution Method page appears.

2. If you specified a .WSI that contains multiple releases, a drop-down list appears. Select a release.

3. Mark **FTP Server**.
If this option is not available, you might not have met the criteria listed at the beginning of this topic.

4. Click Next. If necessary, the installation file is saved and compiled.

The FTP Server page appears.

5. Complete the page:

- **FTP Server Address**
  The address of the FTP server to transfer the package files to. Example: ftp.server.com.

- **FTP Logon Name**
  A valid logon name for this FTP server. The logon name must have write access to the directory to transfer files to.

- **FTP Logon Password**
  A valid password.

- **FTP Upload Directory**
  The directory on the FTP server to copy the package to. The first character must be a forward slash (/). Example: /published/installations.

6. Click Next.

   The package is uploaded to the FTP server. A dialog box shows the status of the upload.

7. Click Finish.

**Note**
If this option does not work as you expect, open an FTP client, configure it with the same information you entered in Package Distribution, and make sure it works. (Windows contains a default FTP client.)

---

**Performing an Administrative Installation of a Windows Installer Package**

When a Windows Installer package is ready to deploy to end users, you can use Package Distribution to perform an administrative installation. An administrative installation copies a source image of the application to a network; the source image resembles the directory structure of the installed application. End users who have access to the administrative installation can then install the application from the network location.

See *Administrative Installation* in the Windows Installer SDK Help.

---

<table>
<thead>
<tr>
<th>You specify this type of file:</th>
<th>.MSI, .WSI</th>
</tr>
</thead>
<tbody>
<tr>
<td>If you specify a .WSI, it is compiled and the compiled installation is distributed.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>You run Package Distribution from:</th>
<th>Workbench</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software Manager</td>
<td></td>
</tr>
</tbody>
</table>

---
To perform an administrative installation

1. Do one of the following:
   - On the Projects tab, click the Run link to the right of the task or tool associated with Package Distribution. The package associated with the current project will be distributed. This tool might skip dialog boxes or pre-fill fields based on command-line options defined in Process Templates Setup.
   - On the Tools tab, double-click Package Distribution. On the Specify File to Distribute page that appears, specify a file and click Next.
   - In Software Manager, in the Applications/Packages pane, select a package. Then select Packages menu > Distribute.
     The Distribution Method page appears.

2. If you specified a .WSI that contains multiple releases, a drop-down list appears. Select a release.

3. Mark **Administrative Installation**.
   - If this option is not available, you might not have met the criteria listed at the beginning of this topic.

4. Click Next. If necessary, the installation file is saved and compiled.
   - The Administrative Installation page appears.

5. Complete the page:
   - **Network Directory**
     Specify the directory in which to place the administrative installation.
   - **Use Short File Names**
     Mark this if you are copying the administrative installation to a location that does not support long file names. This sets the SHORTFILENAME property, causing all file names and directories in the administrative installation to be shortened to their 8.3 equivalent names. See *SHORTFILENAME Property* in the Windows Installer SDK help.

6. Click Finish.
   - An executable version of the package is copied to the directory you specified.

---

Other requirements:

- Does not apply to WiseScript packages.
- Not available if distributing a preflight package (generated with Preflight Instrumentation tool.)
Chapter 10
Preflight Deployment

This chapter includes the following topics:

- About Preflight Deployment on page 259
- The Preflight Deployment Process on page 261
- Connection to Preflight Deployment Tools on page 261
- Creating a Preflight Package on page 262
- Viewing Results from Preflight Deployment on page 263
- Preflight Diagnostic Tests on page 265

About Preflight Deployment

Quality Assurance module only.

Preflight Deployment™ helps you determine if an installation or patch will succeed or fail by testing it in your production environment before deployment. Preflight Deployment lets you discover why installations or patches fail on specific computers without disrupting end users or endangering your production environment.

With Preflight Deployment, you generate a preflight package based on a package that you plan to deploy. Deploy the preflight package using your normal deployment method. When the preflight package runs, it performs tests based on the package contents without actually making any changes on the target computers, then sends the results to a Web server. Use the Preflight Analysis tool to view results.

Parts of Preflight Deployment

- Preflight Instrumentation
  
  The Preflight Instrumentation tool in Workbench reads and analyzes a package that you plan to deploy. It then generates a preflight package based on that package. The preflight package simulates the installation but does not make changes on the target computer. The preflight package:
  
  - Disables all changes to the target computer, such as installation of files, registry entries, and so on.
  - Is stripped of non-Wise custom actions, because they could make changes to the target computer. Because of this, some tests might return results that would differ from the results if the original package ran.
  
  See Interpreting preflight results and troubleshooting on page 264.
  
  - Contains added Wise custom actions that test certain conditions, such as disk space, privileges, and launch conditions. Wise custom actions also gather the results from these tests and send them to a Web application named Preflight Data Collector.
  
  - Is condensed down to the smallest practical size (no .CABs, no graphics, and so on).
See *Creating a Preflight Package* on page 262.

- **Preflight Analysis**
  The Preflight Analysis tool in Workbench is a Web application that shows the results of the preflight package.
  
  See *Viewing Results from Preflight Deployment* on page 263.

- **Preflight Data Collector**
  Preflight Data Collector is a Web application that receives the data from the preflight package, unpacks it, and inserts it into the Wise Services database. This Web application has no user interface.

- **Wise Services Database**
  Stores data from preflight packages and interacts with the Preflight Data Collector and Preflight Analysis Web applications.

See also:

*The Preflight Deployment Process* on page 261
*Connection to Preflight Deployment Tools* on page 261
*Command Line Options for Preflight Instrumentation* on page 280
The Preflight Deployment Process

Quality Assurance module only.

Phase 1:
Use Preflight Instrumentation to create a preflight package based on a package that you plan to deploy.

Phase 2:
Deploy the preflight package to end user computers using your own deployment system.

Phase 3:
Use Preflight Analysis to view results.

Connection to Preflight Deployment Tools

Quality Assurance module only.

For Preflight Deployment to work, the Preflight Web applications must be installed on a Microsoft Internet Information Services (IIS) Web server on your network. You must also ensure that the other requirements for the Preflight Web applications have been met. (See System Requirements in the Wise Package Studio Getting Started Guide.)

The Preflight Web applications handle the data generated from preflight packages that you deploy.

Preflight Data Collector Web application

When you create a preflight package using the Preflight Instrumentation tool, you must specify the URL of the Data Collector Web application. The URL is embedded in the preflight package so that when it runs, it sends its results to the Data Collector Web application. Obtain the Data Collector URL from the team member who installed Preflight Web server.
Preflight Deployment

Web applications. Typically, the URL for the Data Collector Web application would be something like this:

http://IIS_Server/Wise_Managed_Enterprise/wisewebservice.dll

Note
If the package runs on a remote computer, you might need to provide a fully qualified domain name for the IIS server, depending on network and remote computer configuration.

Preflight Analysis Web application

If a team member previously installed the Preflight Web applications, and if you pick the same share point directory that was selected during Preflight Web applications installation, then the Preflight Analysis tool is automatically set up with the correct URL.

If not, then you might need to set up the URL yourself. If so, obtain the correct URL from the team member who installed Preflight Web applications. Then, select Edit menu > Tools, click the Preflight Analysis tool, and enter the complete URL in the URL field.

Typically, the URL for Preflight Analysis would be something like this:

http://IIS_Server/Wise_Preflight_Analysis/

Wise Services database

On the IIS Web server, you can change connection information for the SQL database that the Preflight Web applications interact with. To do so, use the Wise Repository Manager. See Managing the Wise Software Repository in the Getting Started Guide.

For information about installation, see Installing Web Applications in the Getting Started Guide.

Creating a Preflight Package

➤ Quality Assurance module only.

Use the Preflight Instrumentation tool to create a preflight .MSI package from a package that you plan to deploy. You then deploy the preflight package, which tests end user computers before deployment of the real package.

See The Preflight Deployment Process on page 261.

You can create a preflight package from these package types: .MSI, .WSI, hotfix .EXE, and universal import .EXE.

The Preflight Instrumentation tool does not modify the package you choose, but instead makes a special preflight .MSI of the package that does not make any changes to end user computers. The preflight .MSI contains custom actions that check the items listed in Preflight Diagnostic Tests on page 265.

To create a preflight package

1. In Workbench, do one of the following:
   - On the Tools tab, double-click Preflight Instrumentation. In the Original Package dialog box that appears, specify a package to test with Preflight Deployment and click Next. To open a package from the Wise Software Repository, click Browse and then click the Repository tab.
Preflight Deployment

See Opening an Installation Package in the Windows Installer Editor Help.

If you select any type of package other than an .MSI, you should open it from the repository. This means it has to be imported to Software Manager.

- On the Projects tab, click the Run link to the right of the task or tool associated with Preflight Instrumentation. The package that is created will be saved with the default project name plus “_preflight” added to its name. This tool might skip dialog boxes or populate fields based on command-line options defined in Process Templates Setup.

The Preflight Package to Create dialog box appears.

2. Specify a path of the preflight package to create and click Next.

The Data Collector URL dialog box appears. The Data Collector Web application is set up during installation of the Preflight Web applications. If you did not install the Preflight Web applications, you must obtain this URL from the team member who did.

See Connection to Preflight Deployment Tools on page 261.

3. Specify the URL of the Preflight Data Collector Web application and click Next.

The Deployment Job Identifier dialog box appears. A job identifier is used to group a set of results from this preflight package. In Preflight Analysis, you view results by job identifier.

4. Enter a unique job identifier that describes the package you are testing and click Next. (Example: Microsoft Office 2002)

The Package Instrumentation dialog box appears.

5. Click Next, and the preflight package is generated at the path you specified. Click Finish after the preflight package is created.

Note

Package Instrumentation also validates that the package exists in the Software Manager database (Professional Edition only). If one or more files from the package are unmanaged by the database, you are prompted to import the package. If you do so, Package Distribution opens temporarily and prompts you to distribute to the share point directory. You should cycle the imported package through your normal QA and conflict resolution processes.

See Copying a Package to the Share Point Directory on page 252.

The next step in Preflight Deployment is to deploy the preflight package using the deployment system you normally use. This step must be performed independently. After you deploy the preflight package, view the results with Preflight Analysis.

See Viewing Results from Preflight Deployment on page 263.

Viewing Results from Preflight Deployment

➤ Quality Assurance module only.

Use the Preflight Analysis tool to view the results from the deployment of a preflight job and individual packages. During installation, you must have entered a valid URL to a Preflight Analysis Web application.

Start Preflight Analysis in either of the following ways:
On the Tools tab, double-click Preflight Analysis.

OR

On the Projects tab, click the Run link to the right of the task or tool associated with Preflight Analysis.

Preflight Analysis opens. Use the links and buttons provided to view results and navigate.

If Workbench cannot connect to the computer that is hosting the Preflight Analysis Web application, the Connection Failed dialog box appears.

See Connecting to a Web Application on page 79.

Interpreting preflight results and troubleshooting

The results you see in Preflight Analysis were sent from the target computer to the IIS server by the preflight package. If you have unexpected results, there are several possible causes.

- **Statuses of Incomplete**
  An incomplete in the Status column can indicate that the preflight .MSI failed with a fatal error, network communication was interrupted, or the computer was powered off. It could also be that the preflight testing is still in progress.

- **Network Communication Failures**
  If results from preflight packages do not appear in Preflight Analysis, then the target computer might have not have network access to the IIS server. This can be caused by a variety of network configuration problems, including name resolution problems, cabling problems, operating system configuration, and so on. To avoid this problem, you could locate preflight Web applications on the same computer as you use for deployment, because the deployment computer has a history of successful communication with target computers.

- **FullyQualified Domain Names**
  Remote computers might have trouble connecting to the IIS server if they are required to connect using fully qualified domain names. The URL to the Data Collector Web application is embedded in the preflight package by the Package Instrumentation tool, and usually defaults to something like this:
  
  http://IIS_Server/Wise_Managed_Enterprise/wisewebservice.dll
  
  If a fully qualified domain name is required, the URL would look something like this:
  
  http://IIS_Server.company.com/Wise_Managed_Enterprise/wisewebservice.dll

- **Remote computers using Citrix**
  The preflight package cannot communicate over Citrix.

- **Custom Action Removal**
  Custom actions are removed because they could cause changes to the target computer. However, sometimes this may cause results that differ from the original package. Example: Suppose a custom action sets a property based on the contents of the computer. In turn, the property is used in conditions that cause different actions in the package. Because the custom action is removed, the property is not set, and the conditional branches are not followed.

- **Permission Test Failures**
  Permission tests rely on the current user account having permission to read permission data from the registry. If the user account does not, then permission tests cannot be performed.
• **Windows 2003 Server IIS Configuration**
  If you install Preflight Web applications on a Windows 2003 IIS server, make sure IIS 6.0 is configured to allow ISAPI and ASP, which is prohibited by default.

See also:
- *Connecting to a Web Application* on page 79
- *Connection to Preflight Deployment Tools* on page 261
- *About Preflight Deployment* on page 259
- *The Preflight Deployment Process* on page 261
- *Preflight Diagnostic Tests* on page 265

### Preflight Diagnostic Tests

➤ *Quality Assurance module only.*

When you use Preflight Instrumentation to prepare a package for preflight deployment, a new preflight .MSI package is created based on the original package. The preflight package contains custom actions that perform the following tests, which do not make changes to the target computers. After you run the preflight package on target computers, you can see results from these tests in the Preflight Analysis tool.

- **Begin Run**
  Indicates the beginning of the preflight tests.

- **Check In-Use**
  Reports the files in the package that are in-use at on the computer. These could fail to be installed properly because in-use files cannot be replaced. This test is informational only.

- **Connectivity to URL**
  Extracts URLs from any Launch Web Page, Download File From Internet, or Post Data to HTTP Server custom actions that are found in the original package. It puts them in a table, which is processed on the target computer.

- **Disk Space Check**
  Evaluates the disk costing and identifies target computers that don't meet the requirements.

- **End Run**
  Indicates the end of the preflight tests, which means the preflight package was fully processed.

- **File Association Check**
  Checks that any extensions in the preflight package do not overwrite or reassign those already on the target computer.

- **File Searching**
  Cycles through the AppSearch table and evaluates conditions attached to each AppSearch action. This lets you know if the items specified in AppSearch exist on the target computer.

- **File Security - Read**
  Cycles through the files that are installed and queries the target computer for permissions to access the files.

- **File Security - Write**
  Cycles through the files that are installed and queries the target computer for permissions to create and update files.
<table>
<thead>
<tr>
<th>Test</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files Installed</td>
<td>Reports the number of files that would be installed on the computer and also reports the total number of files contained in the package. Example: If a file is slated to be installed, but does not get installed because the same file already exists, then the cause might be the versioning rules that apply to that file. This test is informational only.</td>
</tr>
<tr>
<td>Find Valid Patch Target</td>
<td>Checks if the computer contains any of the valid patch targets that are recorded within the patch. If it fails to find a valid target, this test fails. This test only applies if the package being tested is a patch.</td>
</tr>
<tr>
<td>Launch Conditions</td>
<td>Determines which launch conditions succeed and which fail.</td>
</tr>
<tr>
<td>Local File Execution</td>
<td>Evaluates custom actions that use a command line to run a file that is expected to exist on the target computer.</td>
</tr>
</tbody>
</table>
| Managed File Check         | Not available unless the Professional Edition is installed in addition to the Quality Assurance module. Checks for unmanaged files and returns a warning if any are found. Unmanaged files are files that are not recorded in the Wise Software Repository. Specifically, an unmanaged file is a file on the target computer that has the same name and location, but differing attributes, as a file in the preflight package. The presence of unmanaged files indicates that software outside the packaging process is installed on target computers, which might invalidate previous conflict analysis and installation testing. To resolve this, do one of the following:  
  • Import the package associated with the unmanaged files into the Software Manager database for conflict analysis and installation testing.  
  • Remove the unauthorized software from target computers. |
<p>| .NET Framework             | Determines if the .NET Framework is required by this installation, and if so, if the .NET Framework is installed on the target computer. This test is placed after the launch condition test. Preflight Instrumentation determines if this test is required by checking the MsiAssembly table for any entries with a 0 in the Attributes column. If the test is not required, the Preflight Instrumentation tool does not add it. |
| Open Document Check        | Builds a list of document extensions from any OpenDocument custom actions and saves the list in a table. When the preflight package runs, this test checks the registry for support for each extension. |</p>
<table>
<thead>
<tr>
<th>Patch Existence</th>
<th>Checks if the patch being tested already is already installed on the computer. If it is, this test fails. This test only applies if the package being tested is a patch.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Registry Security - Read</td>
<td>Cycles through the registry entries that are installed and queries the target computer for permissions to access them.</td>
</tr>
<tr>
<td>Registry Security - Write</td>
<td>Cycles through the registry entries that are installed and queries the target computer for permissions to create and update them.</td>
</tr>
</tbody>
</table>
Appendix A
Wise Package Studio Command Line Options

This chapter includes the following topics:

- **About Wise Package Studio command-line options** on page 268
- **Command Line Options for Application Isolation** on page 269
- **Command Line Options for ApplicationWatch** on page 271
- **Command Line Options for ConflictManager** on page 273
- **Command Line Options for Command Line Builder** on page 272
- **Command Line Options for InstallTailor** on page 273
- **Command Line Options for Legacy Setup Conversion** on page 274
- **Command Line Options for Linux Package Editor** on page 276
- **Command Line Options for Mobile Device Package Editor** on page 277
- **Command Line Options for Package Distribution** on page 277
- **Command Line Options for Package Relationships** on page 278
- **Command Line Options for Package Validation** on page 279
- **Command Line Options for Patch Creation** on page 280
- **Command Line Options for Preflight Instrumentation** on page 280
- **Command Line Options for SetupCapture Configuration** on page 280
- **Command Line Options for SetupCapture** on page 281
- **Command Line Options for SOE Snapshot** on page 283
- **Command Line Options for Software Manager** on page 284
- **Command Line Options for Test Expert** on page 285
- **Command Line Options for UpgradeSync** on page 285
- **Command Line Options for Virtual Package Editor** on page 285
- **Command Line Options for Windows Installer Editor** on page 286
- **Command Line Options for WiseScript Package Editor** on page 288

### About Wise Package Studio command-line options

*Not available in Standard Edition.*

You can use command-line options to affect the way a tool or task runs. (Example: You can set an option that causes Windows Installer Editor to open the default project package automatically.) You enter the command-line options when you create a tool in Tool Setup or when you create a task in Process Templates Setup.
Most predefined Workbench tools require a command-line option in order to run. Example: To run ApplicationWatch, you must use the following command-line option:

```
workbench.exe /tool="ApplicationWatch"
```

In general, you should not change the command-line options of a predefined tool, or the tool might not run properly, if at all.

This section lists the command-line options available for predefined Workbench tools. It also contains examples of how you can use the command-line options to run the tool, using variables to provide information required by the tool program.

See Wise Package Studio Variables on page 75.

Note
The variables described here are usable only within Wise Package Studio. (Example: You cannot use them from the Run command.) To run a tool from a command line, you must enter specific paths and file names instead of variables.

---

# Command Line Options for Application Isolation

The following table lists the command-line options you can use with wfwi.exe to run Application Isolation.

See Application Isolation on page 89.

<table>
<thead>
<tr>
<th>Option</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>/d</td>
<td>Required option to run Application Isolation.</td>
</tr>
<tr>
<td>/d &quot;path\file_name&quot;</td>
<td>Run isolation on a specified file. Place the full path (or appropriate variables) of the source file, the file to isolate, within quotation marks after the space delimiter.</td>
</tr>
<tr>
<td>/d=&quot;path\file_name_Isolated.msi&quot;</td>
<td>Specify the name of the updated .MSI file. Place the full path (or appropriate variables) of the updated package output file within quotation marks after the equals sign. Do not use a space delimiter.</td>
</tr>
<tr>
<td>/d1</td>
<td>Isolate using manifests—optimal for installations that run on Windows XP or later.</td>
</tr>
<tr>
<td>/d11</td>
<td>Isolate using manifests. Isolate automatically.</td>
</tr>
<tr>
<td>/d12</td>
<td>Isolate using manifests. Isolate manually.</td>
</tr>
<tr>
<td>/d111</td>
<td>Isolate using manifests. Isolate automatically. Make package compatible with all operating systems.</td>
</tr>
<tr>
<td>/d112</td>
<td>Isolate using manifests. Isolate automatically. Make package compatible with Windows XP or later.</td>
</tr>
<tr>
<td>/d121</td>
<td>Isolate using manifests. Isolate manually. Make package compatible with all operating systems.</td>
</tr>
<tr>
<td>/d122</td>
<td>Isolate using manifests. Isolate manually. Make package compatible with Windows XP or later.</td>
</tr>
<tr>
<td>/d2</td>
<td>Isolate using Windows Installer isolated components.</td>
</tr>
<tr>
<td>/d21</td>
<td>Isolate using Windows Installer isolated components. Isolate automatically.</td>
</tr>
</tbody>
</table>
### Option Results

<table>
<thead>
<tr>
<th>Option</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>/d211</td>
<td>Isolate using Windows Installer isolated components. Isolate automatically. Move files so .EXE and support files are in the same feature.</td>
</tr>
<tr>
<td>/d212</td>
<td>Isolate using Windows Installer isolated components. Isolate automatically. Only isolate .EXEs and support files that are within the same feature.</td>
</tr>
<tr>
<td>/d22</td>
<td>Isolate using Windows Installer isolated components. Isolate manually.</td>
</tr>
<tr>
<td>/d221</td>
<td>Isolate using Windows Installer isolated components. Isolate manually. Move files so .EXE and support files are in the same feature.</td>
</tr>
<tr>
<td>/d222</td>
<td>Isolate using Windows Installer isolated components. Isolate manually. Only isolate .EXEs and support files that are within the same feature.</td>
</tr>
<tr>
<td>/dxxx1</td>
<td>The fourth space in this command-line option corresponds directly to the Repair support for isolated files group box on the Select Isolation Options dialog box. (The “xxx” represents the first three digits, as described above—do not enter “xxx” in this command line.) If you enter 1 as the fourth digit, it has the same effect as clicking Do not add repair support for isolated files on the Select Isolation Options dialog box.</td>
</tr>
<tr>
<td>/dxxx2</td>
<td>The fourth space in this command-line option corresponds directly to the Repair support for isolated files group box on the Select Isolation Options dialog box. (The “xxx” represents the first three digits, as described above—do not enter “xxx” in this command line.) If you enter 2 as the fourth digit, it has the same effect as clicking Install isolated files in their original location on the Select Isolation Options dialog box.</td>
</tr>
<tr>
<td>/dxxx3</td>
<td>The fourth space in this command-line option corresponds directly to the Repair support for isolated files group box on the Select Isolation Options dialog box. (The “xxx” represents the first three digits, as described above—do not enter “xxx” in this command line.) If you enter 3 as the fourth digit, it has the same effect as clicking Install isolated files in the application directory only on the Select Isolation Options dialog box.</td>
</tr>
</tbody>
</table>

### Examples

The following table shows how you would use the options above to run this tool, using variables to provide information required by the program.

See Wise Package Studio Variables on page 75.

### Note

These examples are shown as they would be entered at the command prompt. When you define a command-line option in Tool Setup, you do not need to include the .EXE in the command line.
Command Line Options for ApplicationWatch

The following table lists the command-line options you can use with workbench.exe to run ApplicationWatch.

See ApplicationWatch on page 94.

<table>
<thead>
<tr>
<th>Option</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>/tool=&quot;ApplicationWatch&quot;</td>
<td>Required option to run ApplicationWatch.</td>
</tr>
<tr>
<td>/tgfmt=</td>
<td>Specify the type of installation file to create. Options are:</td>
</tr>
<tr>
<td></td>
<td>• MSI (to create an .MSI or .WSI)</td>
</tr>
<tr>
<td></td>
<td>• WSE (to create a .WSE).</td>
</tr>
<tr>
<td></td>
<td>Place the option after the equals sign. This option is not necessary if the target file is specified.</td>
</tr>
<tr>
<td>/tgt=&quot;path\file_name&quot;</td>
<td>Specify the name of the file to create, which will populate Target Installation on the Specify Target Installation File dialog box. Place the full path (or appropriate variables) of the output file within quotation marks after the equals sign. Do not use a space delimiter.</td>
</tr>
<tr>
<td>/k=&quot;path\file_name&quot;</td>
<td>Watch a specified source file. Place the full path (or appropriate variables) of the source file within quotation marks after the equals sign. Do not use a space delimiter.</td>
</tr>
<tr>
<td>/zi</td>
<td>In the Specify Target Installation File dialog box, select the option to Add/Update Resources in Existing Installation to append or update the resources from the watched in the existing installation instead of overwriting the existing installation.</td>
</tr>
</tbody>
</table>
Examples

The following table shows how you would use the options above to run this tool, using variables to provide information required by the program.

See Wise Package Studio Variables on page 75.

Note

These examples are shown as they would be entered at the command prompt. When you define a command-line option in Tool Setup, you do not need to include the .EXE in the command line.

<table>
<thead>
<tr>
<th>Desired behavior</th>
<th>Example command line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Do not prompt for file name.</td>
<td><code>path\workbench.exe /tool=&quot;ApplicationWatch&quot; / tgt=&quot;[ProjectDir]\[FileName].wsi&quot;</code></td>
</tr>
<tr>
<td>Instead, save the results in a .WSI, using the default project directory and project file name.</td>
<td></td>
</tr>
<tr>
<td>Do not prompt for file name.</td>
<td><code>path\workbench.exe /tool=&quot;ApplicationWatch&quot; /z1 / tgt=&quot;[ProjectDir]\[FileName].wse&quot;</code></td>
</tr>
<tr>
<td>Instead, append the results to an existing .WSE with the default project file name.</td>
<td></td>
</tr>
<tr>
<td>Watch the default vendor installation specified in the project. Place the results in a .WSI, using the default project directory and project file name.</td>
<td><code>path\workbench.exe /tool=&quot;ApplicationWatch&quot; /k=&quot;[VendorPackage]&quot; / tgt=&quot;[ProjectDir]\[FileName] (ApplicationWatch).wsi&quot;</code></td>
</tr>
</tbody>
</table>

Command Line Options for Command Line Builder

The executable that runs Command Line Builder is WICLB.exe. There are no options for running the executable.

See Command Line Builder on page 97.

You can use variables to provide a file to act upon, as shown in the following table.

See Wise Package Studio Variables on page 75.

Note

These examples are shown as they would be entered at the command prompt. When you define a command-line option in Tool Setup, you do not need to include the .EXE in the command line.

<table>
<thead>
<tr>
<th>Desired behavior</th>
<th>Example command line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Do not prompt for the file to be run with the command line. Instead, run the default project file .MSI.</td>
<td><code>path\WICLB.exe &quot;[ProjectDir]\[FileName].msi&quot;</code></td>
</tr>
</tbody>
</table>
Command Line Options for ConflictManager

The following table lists the command-line option you can use with Manager.exe to run ConflictManager.

See About ConflictManager in the ConflictManager Help.

<table>
<thead>
<tr>
<th>Option</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>/D=&quot;DSN_name&quot;</td>
<td>Specify the Software Manager database to use in ConflictManager. Place the DSN name or the [Database] variable within quotation marks after the equals sign.</td>
</tr>
</tbody>
</table>

Examples

The following table shows how you would use the option above to run this tool, using variables to provide information required by the program.

See Wise Package Studio Variables on page 75.

Note

These examples are shown as they would be entered at the command prompt. When you define a command-line option in Tool Setup, you do not need to include the .EXE in the command line.

<table>
<thead>
<tr>
<th>Desired behavior</th>
<th>Example command line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open ConflictManager to the default Software Manager database.</td>
<td>Manager.exe /d=&quot;[Database]&quot;</td>
</tr>
</tbody>
</table>

Command Line Options for InstallTailor

The following table lists the command-line options you can use with wfwi.exe to run InstallTailor.

See InstallTailor on page 105.

<table>
<thead>
<tr>
<th>Option</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>/t</td>
<td>Required option to run InstallTailor.</td>
</tr>
<tr>
<td>/t=&quot;path\file_name.mst&quot;</td>
<td>Specify the name of the file to create. Place the full path (or appropriate variables) of the output .MST file within quotation marks after the equals sign. Do not use a space delimiter.</td>
</tr>
<tr>
<td>/t &quot;path\file_name&quot;</td>
<td>Specify the file to create a transform for. Place the full path (or appropriate variables) of the source .MSI or .MST file within quotation marks after the space delimiter.</td>
</tr>
</tbody>
</table>
Examples

The following table shows how you would use the options above to run this tool, using variables to provide information required by the program.

See Wise Package Studio Variables on page 75.

Note

These examples are shown as they would be entered at the command prompt. When you define a command-line option in Tool Setup, you do not need to include the .EXE in the command line.

<table>
<thead>
<tr>
<th>Desired behavior</th>
<th>Example command line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prompt for an .MSI file to transform and save the transform in a file with the default project file name.</td>
<td>path\wfwi.exe /t=&quot;[ProjectDir][FileName].mst&quot;</td>
</tr>
<tr>
<td>Use the vendor installation specified in the project and create a transform with the default project file name.</td>
<td>path\wfwi.exe /t=&quot;[ProjectDir][FileName].mst&quot; &quot;[VendorPackage]&quot;</td>
</tr>
</tbody>
</table>

Command Line Options for Legacy Setup Conversion

Legacy Setup Conversion runs ConvertIS.exe when it converts an InstallShield installation (.MSI or .EXE), RIPtoMSI.exe when it converts an Altiris RapidInstall Package (.EXE), and Workbench.exe for all other conversions. Because of this, there are three sets of command-line options for Legacy Setup Conversion.

See Legacy Setup Conversion on page 110.

Command-line Options for Workbench.exe

The following table lists the command-line options you can use with Workbench.exe to run Legacy Setup Conversion.

<table>
<thead>
<tr>
<th>Option</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>/tool=&quot;Legacy Setup Conversion&quot;</td>
<td>Required option to run Legacy Setup Conversion. If this is the only option you specify, the Select Source Format dialog box will open, from which you can select all types of conversions, including InstallShield (.MSI or .EXE).</td>
</tr>
<tr>
<td>/srcfmt=</td>
<td>Specify the type of file you are importing to skip the Select Source Format dialog box. This option is not necessary if the source file is specified. Valid source formats are as follows (case-sensitive); place the source format after the equals sign.</td>
</tr>
<tr>
<td></td>
<td>SMS</td>
</tr>
<tr>
<td></td>
<td>ZENWorks</td>
</tr>
<tr>
<td></td>
<td>WinINSTALL</td>
</tr>
<tr>
<td></td>
<td>WiseScript</td>
</tr>
<tr>
<td></td>
<td>InstallShield</td>
</tr>
</tbody>
</table>
**Wise Package Studio Command Line Options**

**Note**
If you specify both the source file and the target file, the Specify Files dialog box is skipped when the tool is run.

---

### Command-line Options for ConvertIS.exe

The following table lists the command-line options you can use with ConvertIS.exe for InstallShield installation (.MSI or .EXE) conversions. ConvertIS.exe is located in the Windows Installer Editor/ConvertIS directory.

<table>
<thead>
<tr>
<th>Option</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>/src=&quot;path\file_name&quot;</td>
<td>Specify the source file of the project to convert, which will populate <em>Source Installation</em> on the Specify Files dialog box. Place the full path and file name (or appropriate variables) within quotation marks after the equals sign.</td>
</tr>
<tr>
<td>/tgfmt=</td>
<td>Specify the type of installation file to create. Options are:</td>
</tr>
<tr>
<td></td>
<td>• <em>MSI</em> (to create an .MSI or .WSI)</td>
</tr>
<tr>
<td></td>
<td>• <em>WSE</em> (to create a .WSE)</td>
</tr>
<tr>
<td></td>
<td>Place the option after the equals sign. This option is not necessary if the target file is specified.</td>
</tr>
<tr>
<td>/tgt=&quot;path\file_name&quot;</td>
<td>Specify the name of the installation file to create, which will populate <em>Target Installation</em> on the Specify Files dialog box. Place the full path and file name (or appropriate variables) within quotation marks after the equals sign.</td>
</tr>
</tbody>
</table>

---

### Command-line Options for RIPtoMSI.exe

The following table lists the command-line options you can use with RIPtoMSI.exe to convert Altiris RapidInstall packages (RIPs) to .MSIs. The RIPtoMSI.exe is located in the Wise Package Studio\Workbench directory.

<table>
<thead>
<tr>
<th>Option</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>/d:&quot;path\file_name&quot;</td>
<td>Specify the name and location of the destination file (.MSI).</td>
</tr>
<tr>
<td>/l:&quot;path\file_name&quot;</td>
<td>Specify the name and location of the log file.</td>
</tr>
<tr>
<td>/s:&quot;path\file_name&quot;</td>
<td>Specify the name and location of the source file.</td>
</tr>
</tbody>
</table>

**Option Results**

<table>
<thead>
<tr>
<th>Option</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>/s</td>
<td>Run Legacy Setup Conversion silently.</td>
</tr>
<tr>
<td>/o</td>
<td>Open the converted .MSI or .WSI in Windows Installer Editor.</td>
</tr>
<tr>
<td>/src=&quot;path\file_name&quot;</td>
<td>Specify the source file of the project to convert, which will populate <em>Source Installation</em> on the Specify Files dialog box. Place the full path and file name (or appropriate variables) within quotation marks after the equals sign.</td>
</tr>
<tr>
<td>/tgt=&quot;path\file_name&quot;</td>
<td>Specify the name of the installation file to create, which will populate <em>Target Installation</em> on the Specify Files dialog box. You can specify an .MSI or .WSI. Place the full path and file name (or appropriate variables) within quotation marks after the equals sign.</td>
</tr>
</tbody>
</table>
Wise Package Studio Command Line Options

### Examples

The following table shows how you would use the options above to run Legacy Setup Conversion, using variables to provide information required by the program.

See [Wise Package Studio Variables](#) on page 75.

#### Note

These examples are shown as they would be entered at the command prompt. When you define a command-line option in Tool Setup, you do not need to include the .EXE in the command line.

<table>
<thead>
<tr>
<th>Desired behavior</th>
<th>Example command line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convert an SMS Installer file into a .WSI with the default project file name.</td>
<td><code>path\workbench.exe /tool=&quot;Legacy Setup Conversion&quot; /tgt=&quot;[ProjectDir]\[FileName].wsi&quot; /srcfmt=SMS</code></td>
</tr>
<tr>
<td>Convert a WinInstall file to a WiseScript (.WSE) and prompt for the source and target file names.</td>
<td><code>path\workbench.exe /tool=&quot;Legacy Setup Conversion&quot; /tgtfmt=wse /srcfmt=WinINSTALL</code></td>
</tr>
<tr>
<td>Silently convert an InstallShield .MSI into a non-proprietary .WSI and open it in Windows Installer Editor.</td>
<td><code>path\ConvertIS.exe /src=[ProjectDir]\[FileName].msi&quot; /tgt=[ProjectDir]\[FileName].wsi&quot; /s /o</code></td>
</tr>
<tr>
<td>Convert an Altiris RIP file to an .MSI with only the Performing Migration dialog box appearing.</td>
<td><code>path\RIPtoMSI.exe /s:[ProjectDir]\[FileName].exe&quot; /d:=[ProjectDir]\[FileName].msi&quot; /u</code></td>
</tr>
</tbody>
</table>

### Command Line Options for Linux Package Editor

The executable that runs Linux Package Editor is WiseForLinux.exe. There are no options for running the executable.

See [About Linux Package Editor](#) in the Linux Package Editor Help.

You can use variables to provide a file to act upon, as shown in the following table.

See [Wise Package Studio Variables](#) on page 75.
Note
These examples are shown as they would be entered at the command prompt. When
you define a command-line option in Tool Setup, you do not need to include the .EXE in
the command line.

## Command Line Options for Mobile Device Package Editor

The executable that runs Mobile Device Package Editor is MDEditor.exe. There are no
options for running the executable.

See About Mobile Device Package Editor in the Mobile Device Package Editor Help.

You can use variables to provide a file to act upon, as shown in the following table.

See Wise Package Studio Variables on page 75.

Note
These examples are shown as they would be entered at the command prompt. When
you define a command-line option in Tool Setup, you do not need to include the .EXE in
the command line.

<table>
<thead>
<tr>
<th>Desired behavior</th>
<th>Example Command-line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open the default project file .LPR.</td>
<td>path\WiseForLinux.exe &quot;[ProjectDir][FileName].lpr&quot;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Desired behavior</th>
<th>Example command line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open the default project file .INF.</td>
<td>path\MDEditor.exe &quot;[ProjectDir][FileName].inf&quot;</td>
</tr>
</tbody>
</table>

## Command Line Options for Package Distribution

The following table lists the command-line options you can use with workbench.exe to run Package Distribution.

See Package Distribution on page 249.

<table>
<thead>
<tr>
<th>Option</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>/tool=&quot;Package Distribution&quot;</td>
<td>Required option to run Package Distribution.</td>
</tr>
<tr>
<td>/srcfmt=</td>
<td>Specify the type of installation file to distribute. Options are:</td>
</tr>
<tr>
<td></td>
<td>• MSI (to create an .MSI or .WSI)</td>
</tr>
<tr>
<td></td>
<td>• WSE (to create a .WSE)</td>
</tr>
<tr>
<td></td>
<td>Place the option after the equals sign. This option is not necessary if</td>
</tr>
<tr>
<td></td>
<td>the source file is specified.</td>
</tr>
</tbody>
</table>
The following table shows how you would use the options above to run this tool, using variables to provide information required by the program.

See [Wise Package Studio Variables](#) on page 75.

### Note
These examples are shown as they would be entered at the command prompt. When you define a command-line option in Tool Setup, you do not need to include the .EXE in the command line.

<table>
<thead>
<tr>
<th>Desired behavior</th>
<th>Example command line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distribute the default project .WSE and prompt for the destination.</td>
<td><code>path\workbench.exe /tool=&quot;Package Distribution&quot; /src=&quot;[ProjectDir]\[FileName].wse&quot;</code></td>
</tr>
<tr>
<td>Distribute the default project .WSI to the share point directory and assign the default project Application and Package names.</td>
<td>`path\workbench.exe /tool=&quot;Package Distribution&quot; /tgt=21=&quot;[SharePoint][ProjectDir][FileName].wsi</td>
</tr>
<tr>
<td>Distribute the default project .WSI to a network directory.</td>
<td><code>path\workbench.exe /tool=&quot;Package Distribution&quot; /tgt=23 /src=&quot;[ProjectDir]\[FileName].wsi&quot;</code></td>
</tr>
</tbody>
</table>

---

**Command Line Options for Package Relationships**

The following table lists the command-line option you can use with Manager.exe to access the Package Relationships dialog box.

See [About Package Relationships](#) in the Software Manager Help.
### Command Line Options for Package Validation

The following table lists the command-line option you can use with wfwi.exe to run Package Validation.

See *About Package Validation* on page 145.

<table>
<thead>
<tr>
<th>Option</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>/g</td>
<td>Required option to run Package Validation.</td>
</tr>
</tbody>
</table>

#### Examples

The following table shows how you would use the option above to run this tool, using variables to provide information required by the program.

See *Wise Package Studio Variables* on page 75.

#### Note

These examples are shown as they would be entered at the command prompt. When you define a command-line option in Tool Setup, you do not need to include the .EXE in the command line.
Command Line Options for Patch Creation

The following table lists the command-line option you can use with wfwi.exe to run Patch Creation.

See Patch Creation on page 128.

<table>
<thead>
<tr>
<th>Desired behavior</th>
<th>Example command line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Validate the default project vendor package.</td>
<td>path\wfwi.exe /g &quot;[VendorPackage]&quot;</td>
</tr>
</tbody>
</table>

Command Line Options for Preflight Instrumentation

The executable that runs Preflight Instrumentation is PkgInst.exe. There are no options for running the executable.

See Creating a Preflight Package on page 262.

You can use variables to provide a file to act upon, as shown in the following table.

See Wise Package Studio Variables on page 75.

Note
These examples are shown as they would be entered at the command prompt. When you define a command-line option in Tool Setup, you do not need to include the .EXE in the command line.

<table>
<thead>
<tr>
<th>Desired behavior</th>
<th>Example command line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Do not prompt for the file to act upon. Instead, run the default project file .MSI.</td>
<td>path\PkgInst.exe &quot;[ProjectDir][FileName].msi&quot;</td>
</tr>
</tbody>
</table>

Command Line Options for SetupCapture Configuration

The following table lists the command-line option you can use with wfwi.exe to run SetupCapture Configuration.


<table>
<thead>
<tr>
<th>Option</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>/r1</td>
<td>Required option to run SetupCapture Configuration.</td>
</tr>
</tbody>
</table>
# Command Line Options for SetupCapture

The following table lists the command-line options you can use with workbench.exe to run SetupCapture.

See *About Capturing Applications* on page 201.

<table>
<thead>
<tr>
<th>Option</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>/tool=&quot;SetupCapture&quot;</td>
<td>Required option to run SetupCapture.</td>
</tr>
</tbody>
</table>
| /tgfmt=        | Specify the type of installation file to create. Options are:  
|                |  - MSI (to create an .MSI or .WSI)  
|                |  - WSE (to create a .WSE)  
|                | Place the option after the equals sign. This option is not necessary if the target file is specified.                                 |
| /tg="path\file_name" | Specify the name of the installation file to create, which will populate Target Installation on the Specify Target Installation File dialog box. Place the full path and file name (or appropriate variables) after the equals sign. |
| /app="application_name" | Populate the application name that is used in the Wise Software Repository with the default application name. Place the application name or the [ApplicationName] variable after the equals sign. |
| /pack="package_name" | Populate the package name that is used in the Wise Software Repository with the default package name. Place the package name or the [PackageName] variable after the equals sign. |
| /k="path\file_name" | Capture a specified installation package. Place the full path of a specific installation (or the [VendorPackage] variable) within quotation marks after the equals sign. |
| /qa="application_name" | Populate Name in the Finish dialog box with the default application name. Place the application name or the [ApplicationName] variable after the equals sign. |
| /qm="vendor_name" | Populate Manufacturer in the Finish dialog box with the default product vendor. Place the vendor name or the [ProductVendor] variable after the equals sign. |
| /z1             | Append the captured information to an existing project installation file. In the Specify Target Installation File dialog box, selects the option to *Add/Update Resources in Existing Installation*. |
| /z2             | Overwrite an existing project installation file, if one exists. Use this if the analysis phase of a process creates an installation file. |
| /src=4="path"  | Use this option with an .MSI or .WSI file. The "4" causes the source files from the captured installation to be copied to a network directory. Place the full path (or appropriate variables) of the directory within quotation marks after the equals sign. |
### Examples

The following table shows how you would use the options above to run this tool, using variables to provide information required by the program.

See [Wise Package Studio Variables](#) on page 75.

### Note

These examples are shown as they would be entered at the command prompt. When you define a command-line option in Tool Setup, you do not need to include the .EXE in the command line.

<table>
<thead>
<tr>
<th>Desired behavior</th>
<th>Example command line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Save the capture results in a .WSI file with the default project file name and location.</td>
<td><code>path\workbench.exe /tool=&quot;SetupCapture&quot; / tgt=&quot;[ProjectDir]\[FileName].wsi&quot;</code></td>
</tr>
<tr>
<td>Save the capture results in a .WSI file with the default project file name and location. Populate the application and package names used in the Wise Software Repository with the default application and package names. Capture the default vendor package. Append the default application name and manufacturer.</td>
<td><code>path\workbench.exe /tool=&quot;SetupCapture&quot; /app=&quot;[ApplicationName]&quot; /pack=&quot;[PackageName]&quot; /k=&quot;[VendorPackage]&quot; / qa=&quot;[ApplicationName]&quot; /qm=&quot;[ProductVendor]&quot; / tgt=&quot;[ProjectDir]\[FileName].wsi&quot;</code></td>
</tr>
</tbody>
</table>

### Option Results

<table>
<thead>
<tr>
<th>Option</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>/src=45=&quot;path&quot;</code></td>
<td>Use this option with an .MSI or .WSI file. The &quot;4&quot; causes the source files from the captured installation to be copied to a network directory. The &quot;5&quot; indicates that the paths to these files should be written as relative paths, relative to the .MSI/.WSI location. Place the full path (or appropriate variables) of the directory within quotation marks after the equals sign.</td>
</tr>
<tr>
<td><code>/src=4=&quot;path&quot;</code></td>
<td>Use this option with a .WSE or .WVP file and be sure to include the pipe character. The &quot;4&quot; causes the source files from the captured installation to be copied to a network directory. Place the full path (or appropriate variables) of the directory within quotation marks after the equals sign.</td>
</tr>
<tr>
<td><code>/src=45=&quot;path&quot;</code></td>
<td>Use this option with a .WSE or .WVP file and be sure to include the pipe character. The &quot;4&quot; causes the source files from the captured installation to be copied to a network directory. The &quot;5&quot; indicates that the paths to these files should be written as relative paths, relative to the .WSE or .WVP location. Place the full path (or appropriate variables) of the directory within quotation marks after the equals sign.</td>
</tr>
</tbody>
</table>
Wise Package Studio Command Line Options

Command Line Options for SOE Snapshot

The following table lists the command-line option you can use with wfwi.exe to run SOE Snapshot.

See SOE Snapshot on page 242.

<table>
<thead>
<tr>
<th>Option</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>/r3</td>
<td>Required option to run SOE Snapshot.</td>
</tr>
</tbody>
</table>

Examples

The following table shows how you would use the option above to run this tool, using variables to provide information required by the program.

See Wise Package Studio Variables on page 75.

Note

These examples are shown as they would be entered at the command prompt. When you define a command-line option in Tool Setup, you do not need to include the .EXE in the command line.
Command Line Options for Software Manager

The following table lists the command-line options you can use with Manager.exe to run Software Manager.

See About Software Manager in the Software Manager Help.

<table>
<thead>
<tr>
<th>Desired behavior</th>
<th>Example command line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Save the results in a file named SOESnapshot.soe in the default project directory.</td>
<td>path\wfwi.exe /r3 &quot;[ProjectDir]\SOESnapshot.soe&quot;</td>
</tr>
</tbody>
</table>

### Option

<table>
<thead>
<tr>
<th>Option</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>/s</td>
<td>Required option to run Software Manager.</td>
</tr>
<tr>
<td>/j=&quot;path/file_name.que&quot;</td>
<td>Import a package into the Software Manager database from the share point directory.</td>
</tr>
<tr>
<td>/x=</td>
<td>Change the status of a package in the Software Manager database. Possible values: 0 = empty; 1 = Under Development; 2 = Available; 3 = Retired. Use the /A, /P, and /D options to specify the package; the order of the options is not important.</td>
</tr>
<tr>
<td>/A=&quot;application_name&quot;</td>
<td>Use with the /x option to specify the application that contains the package to change the status for.</td>
</tr>
<tr>
<td>/P=&quot;package_name&quot;</td>
<td>Use with the /x option to specify the package to change the status for.</td>
</tr>
<tr>
<td>/D=&quot;DSN_name&quot;</td>
<td>Specify the Software Manager database to use in Software Manager. Place the DSN name or the [Database] variable within quotation marks after the equals sign.</td>
</tr>
<tr>
<td>/q</td>
<td>Run Impact and Risk Assessment.</td>
</tr>
</tbody>
</table>

### Examples

The following table shows how you would use the options above to run this tool, using variables to provide information required by the program.

See Wise Package Studio Variables on page 75.

<table>
<thead>
<tr>
<th>Desired behavior</th>
<th>Example command line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import the default project package into the default Software Manager database.</td>
<td>path\Manager.exe /j1=&quot;[Sharepoint]\Scripts[FileName].que&quot; / D=&quot;[Database]&quot;</td>
</tr>
</tbody>
</table>
Command Line Options for Test Expert

The executable that runs Test Expert is TestExpert.exe. There are no options for running the executable.

See About Test Expert on page 159.

You can use variables to provide information required by the program, as shown in the following table.

See Wise Package Studio Variables on page 75.

<table>
<thead>
<tr>
<th>Desired behavior</th>
<th>Example command line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change the status of the default project package, in the default database, to Available.</td>
<td><code>path\Manager.exe /x=2 /A=&quot;[ApplicationName]&quot; /P=&quot;[PackageName]&quot; /D=&quot;[Database]&quot;</code></td>
</tr>
</tbody>
</table>

Command Line Options for UpgradeSync

The following table lists the command line option you can use with wfwi.exe to run UpgradeSync.

See UpgradeSync on page 138.

<table>
<thead>
<tr>
<th>Option</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>/yu</td>
<td>Required option to run UpgradeSync.</td>
</tr>
</tbody>
</table>

**Examples**

The following table shows how you would use the option above to run this tool, using variables to provide information required by the program.

See Wise Package Studio Variables on page 75.

<table>
<thead>
<tr>
<th>Desired behavior</th>
<th>Example command line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run UpgradeSync on the default file name and location.</td>
<td><code>path\wfwi.exe /yu &quot;[ProjectDir]\[FileName].msi&quot;</code></td>
</tr>
</tbody>
</table>

Command Line Options for Virtual Package Editor

The executable that runs Virtual Package Editor is SVSEditor.exe. There are no options for running the executable.

See About Virtual Package Editor in the Virtual Package Editor Help.
You can use variables to provide a file to act upon, as shown in the following table.
See *Wise Package Studio Variables* on page 75.

**Note**
These examples are shown as they would be entered at the command prompt. When you define a command-line option in Tool Setup, you do not need to include the .EXE in the command line.

### Command Line Options for Windows Installer Editor

The following table lists the command-line options you can use with wfwi.exe to run Windows Installer Editor.
See *About Windows Installer Editor* in the Windows Installer Editor Help.

<table>
<thead>
<tr>
<th>Desired behavior</th>
<th>Example command line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open the default project file .WVP.</td>
<td><em>path</em>\SVSEditor.exe &quot;<em>[ProjectDir]</em>[FileName].wvp&quot;</td>
</tr>
</tbody>
</table>

**Option** | **Results**                                                                                                                                 |
---|------------------------------------------------------------------------------------------------------------------------------------------|
/n          | Open Windows Installer Editor and suppress the New Installation File dialog box.                                                           |
/c          | Open Windows Installer Editor and compile the package.                                                                                    |
/e          | Open Windows Installer Editor to Setup Editor.                                                                                        |
/e1         | Open Windows Installer Editor to Installation Expert.                                                                                 |
/app="application_name" | Populate the application name that is used in the Wise Software Repository with the default application name. Place the application name or the [ApplicationName] variable after the equals sign. |
/pack="package_name" | Populate the package name that is used in the Wise Software Repository with the default package name. Place the package name or the [PackageName] variable after the equals sign. |
/jpageset   | Open Windows Installer Editor with a specific set of page groups. You define custom page groups by selecting Pages menu > Customize in Windows Installer Editor. This option must be used in conjunction with the /e1 option. |
**Note** | Place the page set name immediately after the /j and include the accelerator key. Example: If you created a custom set of page groups that displays as Admin, you must include the & symbol on the command line to indicate the accelerator key, as in &Admin. |
/yr1        | Open Windows Installer Editor and execute the Resolve wizard.                                                                                                                                 |
| Append the following to the command line to specify the data source and group: | *DSN*|Group|Isolation method (0-none, 1-isolated components, 2-apppaths)|apppath |
### Examples

The following table shows how you would use the options above to run this tool, using variables to provide information required by the program.

See *Wise Package Studio Variables* on page 75.

### Note

These examples are shown as they would be entered at the command prompt. When you define a command-line option in Tool Setup, you do not need to include the .EXE in the command line.

<table>
<thead>
<tr>
<th>Desired behavior</th>
<th>Example command line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open Windows Installer Editor to Installation Expert and display a custom set of page groups named Admin.</td>
<td><code>path\wfwi.exe /n /e1 /j&amp;Admin</code></td>
</tr>
<tr>
<td>Open Windows Installer Editor to Setup Editor.</td>
<td><code>path\wfwi.exe /n /e</code></td>
</tr>
<tr>
<td>Create or edit a file in the default project directory with the default project file name. Populate the application and package names used in the Wise Software Repository with the default application and package names.</td>
<td><code>path\wfwi.exe /app=&quot;[ApplicationName]&quot; /pack=&quot;[PackageName]&quot; /n &quot;[ProjectDir]\[FileName].wsi&quot;</code></td>
</tr>
<tr>
<td>Edit the default ApplicationWatch package.</td>
<td><code>path\wfwi.exe &quot;[ProjectDir]\[FileName] (ApplicationWatch).wsi&quot;</code></td>
</tr>
<tr>
<td>Edit the default project Uninstall package.</td>
<td><code>path\wfwi.exe &quot;[ProjectDir]\[FileName] (Uninstall).wsi&quot;</code></td>
</tr>
<tr>
<td>Edit the default project vendor package.</td>
<td><code>path\wfwi.exe &quot;[VendorPackage]&quot;</code></td>
</tr>
<tr>
<td>Edit the default project transform file.</td>
<td><code>path\wfwi.exe &quot;[ProjectDir]\[FileName].mst&quot;</code></td>
</tr>
<tr>
<td>Edit the installation template.</td>
<td><code>path\wfwi.exe &quot;[PackageStudioDir]\Windows Installer Editor\Templates\Windows Application.msi&quot;</code></td>
</tr>
<tr>
<td>Edit the merge module template.</td>
<td><code>path\wfwi.exe &quot;[PackageStudioDir]\Windows Installer Editor\Templates\Merge Module.msm&quot;</code></td>
</tr>
<tr>
<td>Open a new project.</td>
<td><code>path\wfwi.exe /n</code></td>
</tr>
</tbody>
</table>

Option Results

Option: `/yr2`

Open Windows Installer Editor and execute the Resolve with Rules command.

Append the following to the command line to specify the data source, group, and rule set:

```
DSN|Group|Rule Name
```

### Option: `/yr2`

Command line option to open the Windows Installer Editor and execute the Resolve with Rules command.

Append the following to the command line to specify the data source, group, and rule set:

```
DSN|Group|Rule Name
```
The following table lists the command-line options you can use with wise32.exe to run WiseScript Package Editor.

See About WiseScript in the WiseScript Package Editor Help.

**Note**
Running wise32.exe without the /pe option opens WiseScript Editor, which is the scripting-only environment.

### Desired behavior

<table>
<thead>
<tr>
<th>Example command line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compile the default project installation.</td>
</tr>
<tr>
<td>path\wfi.exe /c &quot;[ProjectDir][FileName].wsi&quot;</td>
</tr>
<tr>
<td>Create or edit a new merge module in the default project directory with the default project file name.</td>
</tr>
<tr>
<td>path\wfi.exe /n &quot;[ProjectDir][FileName].msm&quot;</td>
</tr>
<tr>
<td>Open the default project in Windows Installer Editor and run the Resolve wizard against all applications in the Software Manager database.</td>
</tr>
<tr>
<td>path\wfi.exe &quot;[ProjectDir][FileName].wsi&quot; /yr1=&quot;Software Manager Database</td>
</tr>
</tbody>
</table>

---

## Command Line Options for WiseScript Package Editor

The following table lists the command-line options you can use with wise32.exe to run WiseScript Package Editor.

See About WiseScript in the WiseScript Package Editor Help.

**Note**
Running wise32.exe without the /pe option opens WiseScript Editor, which is the scripting-only environment.

### Option

<table>
<thead>
<tr>
<th>Option</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>/pe</td>
<td>Open WiseScript Package Editor.</td>
</tr>
<tr>
<td>/n</td>
<td>Open WiseScript Package Editor and suppress the New Installation File dialog box.</td>
</tr>
<tr>
<td>/c</td>
<td>Open WiseScript Package Editor and compile the package.</td>
</tr>
<tr>
<td>/e</td>
<td>Open WiseScript Package Editor to Script Editor.</td>
</tr>
<tr>
<td>/e1</td>
<td>Open WiseScript Package Editor to Installation Expert.</td>
</tr>
<tr>
<td>/j&amp;Admin</td>
<td>Open WiseScript Package Editor to a specific set of page groups. You define custom page groups by selecting Pages menu &gt; Customize in WiseScript Package Editor. This option must be used in conjunction with the /e1 option.</td>
</tr>
</tbody>
</table>

**Note**
Include the accelerator key if it is used in the page set name. Example: If you created a custom set of page groups that displays as Admin, you must include the & symbol on the command line to indicate the accelerator key, as in &Admin.
Examples

The following table shows how you would use the options above to run this tool, using variables to provide information required by the program.

See Wise Package Studio Variables on page 75.

Note

These examples are shown as they would be entered at the command prompt. When you define a command-line option in Tool Setup, you do not need to include the .EXE in the command line.

<table>
<thead>
<tr>
<th>Desired behavior</th>
<th>Example command line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open WiseScript Package Editor to Installation Expert and display a custom set of page groups named Admin.</td>
<td><code>path\wise32.exe /pe /n /e1 /j&amp;Admin</code></td>
</tr>
<tr>
<td>Open WiseScript Package Editor to Script Editor.</td>
<td><code>path\wise32.exe /pe /n /e</code></td>
</tr>
<tr>
<td>Create or edit a file in the default project directory with the default project file name.</td>
<td><code>path\wise32.exe /pe /n &quot;[ProjectDir]\[FileName].wse&quot;</code></td>
</tr>
<tr>
<td>Edit the default ApplicationWatch package</td>
<td><code>path\wise32.exe /pe &quot;[ProjectDir]\[FileName] (ApplicationWatch).wse&quot;</code></td>
</tr>
<tr>
<td>Edit the default project Uninstall package.</td>
<td><code>path\wise32.exe /pe &quot;[ProjectDir]\[FileName] (Uninstall).wse&quot;</code></td>
</tr>
<tr>
<td>Edit the default project vendor package.</td>
<td><code>path\wise32.exe /pe &quot;[VendorPackage]&quot;</code></td>
</tr>
<tr>
<td>Edit the installation template.</td>
<td><code>path\wise32.exe /pe &quot;[PackageStudioDir]\WiseScript Editor\Template\Empty Project.wse&quot;</code></td>
</tr>
<tr>
<td>Compile the default project installation.</td>
<td><code>path\wise32.exe /c &quot;[ProjectDir]\[FileName].wse&quot;</code></td>
</tr>
<tr>
<td>Edit the default project installation.</td>
<td><code>path\wise32.exe /pe &quot;[ProjectDir]\[FileName].wse&quot;</code></td>
</tr>
</tbody>
</table>
Appendix B
Feature Summary

Wise Package Studio is available in two editions, Professional and Standard, each designed to fulfill the needs of a particular type of user. The version you purchase determines what features are available to you. With the Professional Edition, you can add modules that provide additional functionality: Enterprise Management Server and Quality Assurance.

The following table, which intentionally does not mention all features, only summarizes the differences between each edition and module. If a particular feature is not listed, then it is included in all editions. For a more comprehensive list of all features, refer to the Products section of the Wise Web site. For a list of new features and enhancements in the current release, refer to the Release Notes. In Wise Package Studio, select Help menu > Release Notes.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Edition or Module</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Workbench Tools</strong></td>
<td></td>
</tr>
<tr>
<td>Application Isolation</td>
<td>Professional</td>
</tr>
<tr>
<td>ApplicationWatch</td>
<td>Standard or Professional</td>
</tr>
<tr>
<td>Command Line Builder</td>
<td>Standard or Professional</td>
</tr>
<tr>
<td>ConflictManager</td>
<td>Professional</td>
</tr>
<tr>
<td>Impact and Risk Assessment</td>
<td>Professional</td>
</tr>
<tr>
<td>InstallTailor</td>
<td>Standard or Professional</td>
</tr>
<tr>
<td>Legacy Setup Conversion</td>
<td>Standard or Professional</td>
</tr>
<tr>
<td>Linux Package Editor</td>
<td>Professional</td>
</tr>
<tr>
<td>Mobile Device Package Editor</td>
<td>Standard or Professional</td>
</tr>
<tr>
<td>Package Definition</td>
<td>Professional</td>
</tr>
<tr>
<td>Package Distribution</td>
<td>Standard or Professional or Quality Assurance with Professional</td>
</tr>
<tr>
<td>Package Validation</td>
<td>Standard or Professional or Quality Assurance</td>
</tr>
<tr>
<td>Patch Creation</td>
<td>Standard or Professional</td>
</tr>
<tr>
<td>Preflight Analysis</td>
<td>Quality Assurance</td>
</tr>
<tr>
<td>Preflight Instrumentation</td>
<td>Quality Assurance</td>
</tr>
<tr>
<td>SetupCapture</td>
<td>Standard or Professional</td>
</tr>
<tr>
<td>SetupCapture Configuration</td>
<td>Standard or Professional</td>
</tr>
<tr>
<td>SOE Snapshot</td>
<td>Professional</td>
</tr>
<tr>
<td>Feature</td>
<td>Edition or Module</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>--------------------------------------------------------</td>
</tr>
<tr>
<td>Software Manager</td>
<td>Professional or Quality Assurance with Professional</td>
</tr>
<tr>
<td>Test Expert</td>
<td>Quality Assurance</td>
</tr>
<tr>
<td>UpgradeSync</td>
<td>Standard or Professional</td>
</tr>
<tr>
<td>Virtual Package Editor</td>
<td>Professional</td>
</tr>
<tr>
<td>Web Capture Conversion</td>
<td>Professional</td>
</tr>
<tr>
<td>Windows Installer Editor</td>
<td>Standard or Professional</td>
</tr>
<tr>
<td>Wise Task Manager</td>
<td>Professional</td>
</tr>
<tr>
<td>Wise Web Capture</td>
<td>Professional</td>
</tr>
<tr>
<td>WiseScript Editor, accessible from Windows Installer Editor</td>
<td>Standard or Professional</td>
</tr>
<tr>
<td>WiseScript Package Editor</td>
<td>Professional</td>
</tr>
<tr>
<td><strong>Database Support and Management</strong></td>
<td></td>
</tr>
<tr>
<td>Non-relational Workbench database (.DAT format)</td>
<td>Standard only</td>
</tr>
<tr>
<td>SQL Server and SQL Server Express databases</td>
<td>Professional</td>
</tr>
<tr>
<td>Wise Repository Manager</td>
<td>Professional</td>
</tr>
<tr>
<td><strong>Project Management</strong></td>
<td></td>
</tr>
<tr>
<td>Associate projects with processes, providing step-by-step defined tasks</td>
<td>Professional</td>
</tr>
<tr>
<td>Use predefined processes, which provide built-in industry best practices for repackaging</td>
<td>Professional</td>
</tr>
<tr>
<td>Create your own processes, which are shared among the local team</td>
<td>Professional</td>
</tr>
<tr>
<td>Connect to an external Workbench database and access its processes</td>
<td>Enterprise</td>
</tr>
<tr>
<td>Specify a project or process owner</td>
<td>Enterprise</td>
</tr>
<tr>
<td>Project Management tab for projects, which allows overall resource management of a project</td>
<td>Enterprise</td>
</tr>
<tr>
<td>Metrics tab for projects, which allows task level management of a project</td>
<td>Enterprise</td>
</tr>
<tr>
<td>To Do tab for projects, which allows for team member task assignments</td>
<td>Enterprise</td>
</tr>
<tr>
<td>Add and edit tools, including custom tools</td>
<td>Professional or Quality Assurance</td>
</tr>
<tr>
<td>Workbench reports: Process Documentation</td>
<td>Professional</td>
</tr>
<tr>
<td>Workbench reports: advanced project management reports, To Do List, Security Setup report</td>
<td>Enterprise</td>
</tr>
<tr>
<td>Set user and group-based security that can integrate with Windows NT authentication</td>
<td>Enterprise</td>
</tr>
<tr>
<td>View projects assigned to current user</td>
<td>Enterprise</td>
</tr>
<tr>
<td>Management reports</td>
<td>Enterprise with Management Reports</td>
</tr>
<tr>
<td><strong>ApplicationWatch</strong></td>
<td></td>
</tr>
<tr>
<td>Record watched files in a Windows Installer package (.WSI/.MSI)</td>
<td>Standard or Professional</td>
</tr>
<tr>
<td>Record watched files in a WiseScript (.WSE)</td>
<td>Professional</td>
</tr>
<tr>
<td>Feature</td>
<td>Edition or Module</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>----------------------------------------</td>
</tr>
<tr>
<td>Edit items to be included and excluded from the package resulting from the ApplicationWatch</td>
<td>Professional</td>
</tr>
<tr>
<td><strong>Legacy Setup Conversion</strong></td>
<td></td>
</tr>
<tr>
<td>Convert WinINSTALL installations to a Windows Installer package (.WSI/.MSI)</td>
<td>Standard or Professional</td>
</tr>
<tr>
<td>Convert WinINSTALL installations to a WiseScript (.WSE)</td>
<td>Professional</td>
</tr>
<tr>
<td><strong>ConflictManager</strong></td>
<td></td>
</tr>
<tr>
<td>ConflictManager tool, including rules-based and wizard-based conflict resolution between applications</td>
<td>Professional</td>
</tr>
<tr>
<td>Customize conflict resolution rules</td>
<td>Professional</td>
</tr>
<tr>
<td>Open multiple databases at the same time</td>
<td>Enterprise</td>
</tr>
<tr>
<td>Use groups to reduce the number of conflicts you view at one time</td>
<td>Professional</td>
</tr>
<tr>
<td>Reports</td>
<td>Professional</td>
</tr>
<tr>
<td><strong>Package Distribution tool</strong></td>
<td></td>
</tr>
<tr>
<td>Distribute a Windows Installer package (.WSI/.MSI), merge module, or transform</td>
<td>Standard or Professional</td>
</tr>
<tr>
<td>Distribute a WiseScript (.WSE/.EXE)</td>
<td>Professional</td>
</tr>
<tr>
<td>Distribute to the share point directory</td>
<td>Professional</td>
</tr>
<tr>
<td><strong>Package Validation tool</strong></td>
<td></td>
</tr>
<tr>
<td>Turn individual rules on and off</td>
<td>Standard or Professional or Quality Assurance</td>
</tr>
<tr>
<td>Create a new validation module (.CUB file)</td>
<td>Quality Assurance</td>
</tr>
<tr>
<td>Create new validation rules</td>
<td>Quality Assurance</td>
</tr>
<tr>
<td><strong>SetupCapture</strong></td>
<td></td>
</tr>
<tr>
<td>SetupCapture tool with the ability to capture to a Windows Installer package (.WSI/.MSI)</td>
<td>Standard or Professional</td>
</tr>
<tr>
<td>SetupCapture tool with ability to capture to a WiseScript (.WSE) or virtual software project (.WVP)</td>
<td>Professional</td>
</tr>
<tr>
<td>SetupCapture SmartMonitor or snapshot technology</td>
<td>Standard or Professional</td>
</tr>
<tr>
<td>Perform captures in a virtual software layer</td>
<td>Professional</td>
</tr>
<tr>
<td>Append results of a capture to an existing package</td>
<td>Standard or Professional</td>
</tr>
<tr>
<td>Maintain a global exclusion list, which specifies items to always be ignored during capture</td>
<td>Standard or Professional</td>
</tr>
<tr>
<td>Edit items to be included and excluded from the package resulting from a SetupCapture</td>
<td>Professional</td>
</tr>
<tr>
<td>Edit items to be included and excluded from the package resulting from the SetupCapture</td>
<td>Professional</td>
</tr>
<tr>
<td>Report of installation sequence of events that occur during SetupCapture</td>
<td>Professional</td>
</tr>
<tr>
<td>Report of installation changes performed by the installation during SetupCapture</td>
<td>Professional</td>
</tr>
</tbody>
</table>
Feature Summary

<table>
<thead>
<tr>
<th>Feature</th>
<th>Edition or Module</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add current user registry keys to a separate feature during SetupCapture</td>
<td>Standard or Professional</td>
</tr>
<tr>
<td>Virtual Capture, which creates a Virtual OS file of a clean machine, allowing you to perform SetupCaptures on a non-clean machine.</td>
<td>Professional</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Software Manager</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Revision Control—protect and track packages.</td>
<td>Professional</td>
</tr>
<tr>
<td>Package subscription—copy (subscribe to) packages from one Software Manager database to another. Refresh subscriptions and break subscription links.</td>
<td>Enterprise</td>
</tr>
<tr>
<td>Import all tables from a Windows Installer project; query the tables</td>
<td>Professional</td>
</tr>
<tr>
<td>Import virtual software archive (.VSA) and virtual software project (.WVP) files</td>
<td>Professional</td>
</tr>
<tr>
<td>Import device driver (.INF) information</td>
<td>Professional</td>
</tr>
<tr>
<td>Import Group Policy Object (GPO) information</td>
<td>Professional</td>
</tr>
<tr>
<td>Import an InstallShield® Developer (version 7 or 8) executable</td>
<td>Professional</td>
</tr>
<tr>
<td>Import an installation .EXE of any type</td>
<td>Professional</td>
</tr>
<tr>
<td>Use the Auto Import Service to import packages automatically</td>
<td>Enterprise</td>
</tr>
<tr>
<td>Open multiple databases at the same time</td>
<td>Enterprise</td>
</tr>
<tr>
<td>Group packages</td>
<td>Professional</td>
</tr>
<tr>
<td>Reports</td>
<td>Professional</td>
</tr>
<tr>
<td>Wise Task Scheduler</td>
<td>Enterprise</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Windows Installer Editor</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>WiseScript Editor</td>
<td>Standard or Professional</td>
</tr>
<tr>
<td>ExpressBuild (distributed compile)</td>
<td>Professional</td>
</tr>
<tr>
<td>Resolve conflicts from within Windows Installer Editor</td>
<td>Professional</td>
</tr>
</tbody>
</table>
Symbols
_VistaProtectedFiles, rebuilding 157
_VistaProtectedRegKeys, rebuilding 157
Numerics
000 directory 25, 27
A
action
  in validation rule 154
Active Directory, distributing to 250
Active Project list
  about 19
  filtering 52
  selecting project 19
Admin user 41
administrative installation
  during patching 131
  errors during 131
  performing 257
advertisement, capturing registry info 207
application
  command-line options 63
  defined 14
  executing from SetupCapture 228
  isolating 90
  name 56
  run by tool 67
  running as task 63
  running from
    ApplicationWatch 96
    running from SetupCapture
      Configuration 204
    watching 94
application execution tests
  about 187
  running 189
Application Isolation
  about 89
  command-line options 269
  not available 78
  running 90
Application Monitor
  about 188
  extra files 191
  extra registry entries 191
  file coverage 192
  maximizing 190
  registry coverage 193
Application Specification for Windows 2000 156
Application Specification for Windows XP 156
application verification tests 180
ApplicationName variable 75
ApplicationWatch 94
  command-line options 271
  exclusions 95, 96
  inclusions 96
  not available 78
  running 95
  running application 96
assembly
  patching 129
assigning user to task 81
Assignments by User report 85
Assignments tab
  See Project Management tab
Authenticate 135
automatic task completion 74, 77
Available Packages directory 25
AXT file, converting to Windows Installer 113
B
backup, creating during save 52
baseline machine 218
  for SOE Snapshot 218
  for Test Expert 168
best practices 59
C
C++ symbol file directory 134, 134
CAB file, mobile device 31
Cancel button, hiding in install 100
CAPCOM 135
capture methodology 208
capturing an application
  See SetupCapture
CER file 135
collection
  Internet Connection, test case 178, 178
citrix
  validation 157
    with Preflight 264
Class IDs, test case 181
clean machine
  for SetupCapture 218
  for Test Expert 168
command line
  creating 98
    in package definition 125
in SetupCapture 228
Command Line Builder
  advertising option 102
  command-line options 272
  logging option 101
  not available 78
  patches 104
  public property, changing 104
  repair option 103
  running 98
  transform 104
  UI option 100
command-line options
  about 72
  advertising 102
  Also see individual tool names
application 63
  defining for tool 73
EXE 63
  guidelines for entering 72
  logging 101
  patches, applying 104
  patches, removing 104
  predefined tool 63
  public property, changing 104
  repair 103
  tool 68, 73
  transform, applying 104
  UI options 100
  using with .MSI 97
comparison scan, SetupCapture 225
compile error when using tools 87
compile, on starting tool 64
complete task automatically 74, 77
correlation concepts, Wise Package Studio 13
correlation
  in validation rule 154
configuration file
  Also see SetupCapture
    Configuration
    default 202, 202, 205, 205
    in directory 205
    location 205
  Machine Capture 169, 202
  on local computer 202, 205
  on share point 202, 205
  selecting 205
  SetupCapture 202, 224
  sharing 205
  SOE Snapshot 202, 243
configuration settings
  applying to current capture
    only 220, 224, 244
ConflictManager
about 88
custom action, calling with validation rule 150
custom installation 235
Customize MSI using transform, process 60

darice.cub 156
Database Connectivity, test case 179, 180
database variable
  See SoftwareManagerDSN variable
database, Wise Package Studio 24
directories
  Wise Package Studio 24
directories to watch 208
directory, excluding from SetupCapture 212
distribution method 250
Distribution Wizard
  See Package Distribution
DLL file
calling with validation rule 151
isolating 89
documentation, Wise 33
domain logon 18
domain, Windows 42
download redistributables 232
drive imaging software 218, 227

e
Edit Registry script action, SetupCapture 217
Edit with Word tool 71
Edit with WordPad tool 71
editions, Wise Package Studio 290
  Professional 13, 290
  Standard 13, 290
elapsed time 83
Empty Project.wse 37
Enterprise Management Server 13
environment variable, converting for WinINSTALL 114
error
  found by UpgradeSync 140
  Patch Creation 131
  validation 147
evaluation serial number 50
excluded file, remove from package 204
exclusion
  adding to permanent list 229
  Also see exclusion list
  ApplicationWatch 95, 96
  capturing 203, 210
directory 212
  file 211
  hard-coded 247
  INI file 215
  package definition 127
  registry 214
  SetupCapture 209
  SOE Snapshot 209, 245
types 211
  using wildcard 213
exclusions report, SetupCapture 229, 230
exclusion list
  about 209
  Also see exclusion list
  building automatically 210
  editing 211, 229
guidelines 209
  hard-coded exclusions 247
  items ignored 247
  using to remove files 204

eXE file
calling with validation rule 151
command-line options 63

calling installation 110
Created Files, test case 195
Created Registry Entries, test case 196
credentials file 135, 136
CUB file
  about 148
  Also see validation module customizing 148
class action, calling with validation rule 150
custom installation 235
Customize MSI using transform, process 60

D
darice.cub 156
Database Connectivity, test case 179, 180
database index 27
Database variable
  See SoftwareManagerDSN variable
database, Wise Package Studio 24
database, Software Manager
  See Software Manager database
database, Wise Software Repository
  See Software Manager database
database, Workbench
  See Workbench database
Description tab
  in Projects tab 19
  in Tools tab 21
Destroyed Files, test case 197
Destroyed Registry Entries, test case 198
Details tab 19
dialog box
  hiding with transform 107, 108
digital signature
  adding to patch 135
digital signature, for assembly 92
directories
  Wise Package Studio 24
directories to watch 208
directory, excluding from SetupCapture 212
distribution method 250
Distribution Wizard
  See Package Distribution
DLL file
  converting to Windows Installer 116
defined 31
isolating 89
running as task 63
execute installation from SetupCapture 228
export
  process 65
external Workbench database
  about 60
  copying process 54, 65
Extra Files, test case 190
Extra Registry Entries, test case 191
extract directory 111, 117

F
family, patch 129
features
  advertisement options 236
  attributes 235
  created by SetupCapture 217
  displaying to end user 235
  installation location, setting 235
  requiring 236
features, Wise Package Studio 290
file
  adding from repository 144, 206, 232
  capturing replacement of 225
  excluding from SetupCapture 211
  name, setting for project 56
  verifying existence 63
file and folder exclusions 211
File Coverage, test case 191
file exclusions, in Machine Capture 170
File Extensions, test case 182
file types 31
FileName variable 75
Files in Merge Modules 232
files in repository 144, 206, 232
first use settings, capturing 220, 240
FTP server, distributing to 256
FTP, passive 256
full screen mode 22, 52

G
Getting Started Guide 34
GetWRPItems.exe 157
Global Assembly Cache
  patching 129
global database
  See external Workbench database
gray
  process names 59
task names 59
group, security
  assigning user 42
  creating 39
  defined 39
  from NT domain 39
  importing from NT domain 42
  permissions 40
  predefined 41
  Unassigned 41
  why users get moved 41, 51
  Wise Users 41
  WPS Administrator 41

H

hard drive imaging 218, 227
hardware registry entry, capturing 206
header, task type 62
help
  about 33
  using 33
  Windows Installer SDK 34
Help Files, test case 183
help text
  editing with Word 71
  editing with WordPad 71
  file location 26, 70
  inserting object 71
  task 64, 70
  tool 68, 70
  using .RTF 70
  using HTML 70
  variables 71
hide dialog boxes with transform 107, 108
Hide link 78
History tab
  See Metrics tab
hours completed 81
HTML help text 70
HTTP protocol 256

I

ICE
  defined 149
  editing 155
  errors 147
icon, selecting for tool 68
imaging software 218, 227
Impact and Risk Assessment 88
importing process 66
inclusion
  ApplicationWatch 96
  SetupCapture 228, 241
  SOE Snapshot 244
inclusions report, SetupCapture 229, 230
index, database 27
INF file, mobile device 31
INI file
  capturing as regular file 229
  excluding from SetupCapture 215
initial scan, SetupCapture 227
Initial Workbench Setup
tasks described 35
  using 38
Insert Object tool 71
Install As, in Test Expert 175
installation
  capturing, see SetupCapture changing at run time 105
  converting 110
  creating from installed application 94
  custom 235
  defined 14
  distributing 249
  executing for SetupCapture 228
  monitoring 225
  template, editing 36, 37
  validating 146
  watching 94
Installation Changes report 208
Installation Quality Assurance, process 60
Installation Sequence report 207
installation tests
  about 173
  not appearing 173
  running 174
INSTALLLEVEL 234
InstallShield .MSI
  conversion log 121
  converting 119
  selecting parts to convert 121
InstallShield Pro, converting to Windows Installer 118
InstallTailor
  about 105
  captured changes, about 106
  captured changes, editing 108
  command-line options 273
  not available 78
  running 107
IPF file
  converting to Windows Installer 110
  importing 110
IPR file, converting to Windows Installer 118
Isolated Components report 229
isolated components 91, 94
isolated Files, test case 192
isolation
directory 93
method 91
of applications 90
on Windows XP 93
options 94
OS compatibility 93
report 92
selecting files to isolate 92
self-repair of isolated file 94
type 91
with isolated components 91, 94
with manifests 91, 93

L

Launch Conditions, test case 176
launch Wise Package Studio 16
Legacy Setup Conversion
  about 110
  command-line options 274
  converting InstallShield 118
  converting InstallShield .MSI 119
  converting Novell 113
  converting RapidInstall package 122
  converting SMS 110
  converting WinINSTALL 115
  converting WiseScript 116
  not available 78
  Novell conversion guidelines 112
  WinINSTALL conversion guidelines 114
licenses
  about 47
  adding 49
  Also see serial numbers
  assigning 50
  assigning at logon 48
  deleting 51
  file (.WLC) 49
  unassigning 51
Linux
  installation shell 32
  project 31
Linux Package Editor
  about 88
  command-line options 276
LOCAL file 91
logging option, in command line 101
logo certification
  Windows 2000 156
  Windows Vista 157
logo.cub 156
logon
  current Windows NT account 18
  if logon fails 17
  name 42
  network 18
  options 17
  Windows NT domain 18
  Workbench account 17
LPR file 31

**M**

Machine Capture
- configuration file 169, 202
- directories to watch 169
- during installation test 174
- during uninstall 193
- file exclusions 170
- files ignored 247
- options 169
- registry exclusions 171
- settings 169
- during installation 174
- during uninstall 193
- file exclusions 170
- files ignored 247
- options 169
- registry exclusions 171
- settings 169

Major upgrade 140
MAKECAB.EXE 128
manage projects 80
managed operations 141
Management Reports Web application
- URL 86
- using 86
- manifest, using to isolate 91, 93
- manual, reference 34
- manual, task type 62
- Mark 258
- Master Test Plan, Test Expert 161
- measured time 83
- merge module
  - adding 232
  - defined 32
  - project file 32
  - template, editing 36
- Merge Module.msm 37
- mergemod.cub 156
- Metrics tab 83
- Microsoft Application Specification 156
- Microsoft SMS 110
  - converting to Windows Installer 110
- Minor upgrade 140
- mobile device installation 31
- Mobile Device Package Editor
  - about 88
  - command-line options 277
  - not available 78
- modules, Wise Package Studio 290
- monitor installation 225
- MSI file
  - cached 160
  - converting from InstallShield 119
  - defined 31
  - editing 88
  - testing 159
  - why not repackaging 15
- MSI to WSI Conversion 89
- MSI file
  - about 246
  - decrypting 141
- MSM file
  - Also see merge module defined 32
- MSP file
  - Also see patch defined 32
  - how created 128
- MSPATHC.DLL 128
- MST file
  - Also see transform creating 107
  - defined 32
  - multi-patch 132
- multiple Workbench databases copying process 54, 65

**N**

NAI file, converting 115
navigation, Workbench 19
network directory, distributing to 254
network logon 18
new features
  - Refer to Release Notes
  - notes
    - in Metrics tab 84
    - process 61
    - project 57
- Notes variable 75
- Novell ZENworks
  - converting to Windows Installer 113
- object, in help text 71
- OCX file
  - isolating 89
- ODBC Data Sources, test case 184
- ODBC, capturing 207
- OnDemand Software 114
- operating system
  - clean 218
  - service pack 218
- OS Conditions, test case 176
- OS Snapshot
  - See SOE Snapshot
  - other EXE
    - task type 63
    - tool type 68
  - owner
    - process 61
    - project 56

**P**

package
  - defined 14
  - name 56
  - status 33
  - testing 159
- Package Code 128
- package definition
  - about 123
  - creating 124
  - exclusions 127
  - ignoring files 127
  - importing to Software Manager 123
  - in Software Manager 123
  - in Workbench 123
- package definition file
  - creating 123
  - defined 32
- Package Distribution
  - about 249
  - administrative install 257
  - and relative paths 254
  - choosing method 250
  - command-line options 277
  - distributing to Active Directory 250
  - distributing to FTP server 256
  - distributing to network 254
  - distributing to share point 252
  - not available 78
  - running in Software Manager 249
- Package drop-down list, Test Expert 166
- package relationships
  - command-line options 278
- Package Validation
  - about 145
  - Also see validation
    - command-line options 279
    - correcting problems 147
    - not available 78
    - running 146
- PackageName variable 75
- PackageStudioDir variable 75
- packaging server 169
- packaging, about 14
- parent feature 234
- passive FTP 256
- password
  - for service 245
  - Wise Package Studio 42
- patch
  - advanced settings 137
  - applying to installation 104
  - assembly in GAC 129
  - compile settings 132
  - creating 130
  - defined 32
  - defining upgrade versions 134
  - family 129
  - file 32
  - GUID 134
  - guidelines 128
preparing for 138
project file 32
removing from destination computer 138
removing from installation 104
sequence, specifying 136
uninstalling 138
version to patch 133

Patch Creation
command-line options 280
error on admin install 131
not available 78
previous version, specifying 133
running 130

patch sequence
about 129
specifying 136

PATCHWIZ.DLL 128
PCP file
Also see patch defined 32
how created 128
permissions
Also see security
ConflictManager 43
levels 40
setting 39
setting on database 46
SetupCapture 44
Software Manager 43

Personal Information Exchange
file 135
PFX file 135
predefined application
task type 63
tool type 67
predefined process 59
preferences
general 52
repository 53
SetupCapture 202
SOE Snapshot 202
Software Manager 37
Test Expert 161
Windows Installer Editor 37
WiseScript Package Editor 37
Workbench 52

Preflight
about 259
connection issues 261
data collector 261
file types supported 262
overview 261
package, creating 262
package, deploying 263
parts 259
requirements 261
security tests 264
tests 265
troubleshooting results 264
URLs 262
using Citrix 264
viewing results 263
Preflight Analysis
about 263
not available 78
Preflight database
See Wise Services database
Preflight Instrumentation
about 262
command-line options 280
not available 78
previous version
specifying for patch 133
printing report 86
priority, project 80
private key 136
private key file 135
process
adding task 62
assigning to project 57
best practices 59
copying task 62
creating 61
defined 13
deleting 65
duplicating 65
editing 61
editing in project 57
exporting to file 65
from external database 54, 65
gray font color 59
importing from file 66
multiple releases 59
name 61
notes 61
owner group 61
predefined 59
rearranging 66
restricting changes 61
Process Documentation report 85
Process Templates Setup
about 58
Also see process
interface described 58
predefined process 59
using 58
product vendor 56
ProductVendor variable 75
Professional edition 13, 290
ProgIDs, test case 184
project
assigning a process 57
creating 55
deleted 13
deleting 58
directory 25, 55
duplicating 57
editing 55
elapsed time 83
file location 25, 55
file name 56
history 83
hours completed 81
information 80
managing 80
measured time 83
metrics 83
name 55
owner 56, 80
priority 80
remaining hours 81
restricting changes 56
reverting changes 57
selecting 19
status 33, 55
status, updating automatically 64
with process 20
without process 20, 76
Project Details report 85
project file
merge module 32
patch 32
Project Management tab
about 80
assigning users 81
entering project information 80
entering task time 82
Project Overview report 85
Project Setup
about 54
Also see project
creating project 55
Project Variances report 85
project, Windows Installer 32
ProjectDir variable 75
ProjectName variable 75
Projects directory 25
Projects tab
about 19
interface described 19
restricting access 40
using 76
with process 20
without process 20, 76
prompt, reactivating 53, 161
public property, changing 104
Public/private key pair 135
PVK file 135
Q
QUE file 25
resetting 253
R
RapidInstall package
conversion guidelines 121
converting 122
readme
   See release notes
rebooting from SetupCapture
Configuration 204
reference manual 34
registry
   converting to advertising 207
   excluding from SetupCapture 214
hardware entry, capturing 206
Registry Coverage, test case 193
registry exclusions, in Machine
Capture 171
release notes 34
remaining hours 81
remove excluded file 204
Repackage for Windows Installer,
   process
      about 59
      with multiple releases 59
Repackage into .VSA format,
   process 60
Repackage using WiseScript,
   process 60
repackage.ini
   about 202
   basic exclusion list 247
   using 205
repackaging
   about 14
   with Projects tab 76
   with Tools tab 78
repair option, in command line 103
report
   generating 86
   isolation 92
   predefined 85
   printing 86
   SetupCapture 207, 208, 229, 230
   Workbench 85
report viewer 86
Reports directory 25
repository
   See Wise Software Repository
repository database
   See Software Manager database
Requests tab
   restricting access 40
Residual Files, test case 198
Residual Registry Entries, test
case 199
restarting from SetupCapture
Configuration 204
results file, in Test Expert 163
revert
   project changes 57
revision control, effect on tools 89
RIP
   conversion guidelines 121
   converting 122
RTF help text 70
rule
   See validation rule
rule set, validation
   about 152
   creating 153
   editing 155
run application, from
   ApplicationWatch 96
Run link
   task 20, 77
tool 20, 78
running application
   from ApplicationWatch 96
   from SetupCapture
   Configuration 204
S
Scripts directory 25
Search Locations, test case 185
security
   about 38
   ConflictManager 43
database-level 46
   group 39
   setting 38
   SetupCapture 44
   Software Manager 43
   user 41
   Windows NT authentication 39
security group
   See group
Security Setup report 85
self-repair, isolated file 94
sequence, patch
   about 129
   specifying 136
serial numbers
   about 47
   adding 49
   Also see licenses
deleting 51
evaluation 50
server-side service
   performing operations 143
   user account 143
service pack, on clean machine 218
   service, requiring password 245
Services, test case 186
setting up Wise Package Studio 35, 38
SetupCapture
   about 216
   adding directory 208
   Also see SetupCapture
   Configuration
   baseline machine 218
   buttons disabled 45
clean machine 218
command-line options 281
color items to exclude 209
directory 202, 224
directories to watch 208
directory, excluding 212
Edit Registry script action 217
exclusion list 209
exclusions 229
executing installation 228
   feature, creating 217, 233
   file, excluding 211
files ignored 247
first use settings 220, 240
guidelines 216
inclusions 228, 241
INI file, excluding 215
initial scan 227
method 208
monitoring 225
MSI, capturing 217
multiple installations,
capturing 228
not available 78
ODBC, capturing 207
permissions 44
previous scan 227
registry, converting 207
registry, excluding 214
report 207, 208, 229, 230
root changes, capturing 206, 209
running 219
security settings 44
SmartMonitor 225
snapshot comparison 225
standards, defining 36
summary information 230
uninstall, capturing 217
virtual layer 219
wildcard, excluding 213
SetupCapture Configuration
   about 202
   command-line options 280
directories to watch 208
directory, excluding 212
   file, excluding 211
genral settings 205
INI file, excluding 215
not available 78
registry, excluding 214
running 203
SH file 32
share point directory
   about 26
   changing default 53
   connecting 53
distributing to 252
editing contents 24, 27
example 28
subdirectories 24
temporary file 25, 27
where to locate 26
Sharepoint variable 75
shortcut
for transform 108
Shortcuts, test case 187
Show link 78
side by side mode 22
side-by-side assembly 93
signcode.exe 135
signtool.exe 135
Small update 140
SmartMonitor 225
SMS file
copying 255
snapshot
See SOE Snapshot
snapshot comparison,
SetupCapture 225
SOE file 32
SOE Snapshot
about 242
Also see SetupCapture
Configuration
baseline machine 218
buttons disabled 45
command-line options 283
configuration file 202, 243
exclusion list 209
exclusions 245
files ignored 247
guidelines 243
inclusions 244
running 244
virtual memory 243
Software Manager
about 88
command-line options 284
not available 78
permissions 43
restricting access 43
security settings 43
Software Manager database
about 23
setting permissions 46
Software Virtualization Agent 29
Software Virtualization Solution 29
SoftwareManagerDSN variable 75
source file
indexing 27
location in share point 25
SPC file 135
Standard edition 13, 290
standard operating environment
capturing for conflict analysis 242
capturing, See SOE Snapshot
standard tests 177
standards
company repackaging 35
SetupCapture 36
start Wise Package Studio 16
status, package 33
status, project
about 33
changing 77, 80
setting 55
updating automatically 64, 77
status, task 84
stay on top mode 22
subtask, creating 66
SVS
See Software Virtualization Solution
system requirements, Wise product
Refer to Getting Started Guide
T
task
actual hours 82
adding to process 62
adding tool 63
assigning user 81
completing 20, 77
completing automatically 74, 77
copying 62
creating subtask 66
defined 13
entering time 82
failed status 84
help 64, 70
hierarchy 66
name 62
not available 77
rearranging 66
running 20, 77
status 84
type 62
Wise Task Manager 141
TaskFiles directory 26
template
merge module, editing 36
Windows Installer, editing 36
WiseScript, editing 37
temporary package file 25, 27
Terminal Server validation 157
terminology, Wise Package Studio 13
test case
about 164
adding 172
list of 172
running 165
setting status 166
user-defined 172
Test Expert
about 159
editing test details 166
files ignored 247
in virtual software layer 164
Machine Capture settings 169
not available 78
on multiple computers 168
opening group 166
Package drop-down list 166
results file 163
running test cases 165
setting statuses 166
starting 160
test case list 172
test environment 168
test group
about 161
test item
about 161
setting status 166
text search, in report 86
To Do List report 85
To Do tab
about 84
adding item 84
tool
adding 67
associating with task 63
command-line options 63, 68, 73
defined 14
deleting 69
duplicating 69
editing 67
full screen mode 52
help 68, 70
hiding 78
hiding in Tools tab 68
minimizing 78
not available 78
restricting access 40
revision control interaction 89
running 20, 78
selecting icon 68
showing 78
tool configuration 73
Tool Setup
about 66
Also see tool
using 66
Tools tab
about 21
Description tab 21
interface described 21
rearranging 70
restricting access 40
using 78
when to use 78
transform
applying to installation 104
creating 107
file 32
tutorial
Refer to Getting Started Guide

Unassigned group 41
uninstall
patch 138
uninstall tests
about 193
not appearing 194
running 194
with group 166
uninstall, capturing 217
upgrade, preparing for 138
UpgradeSync 138
command-line options 285
not available 78
running 139
upgrading
errors, avoiding 138
patch, creating 130
specifying for patch 134
specifying version for 133

URL, Management Reports 86
user
Admin 41
creating 41
deleted 41
group assignment 42
name, Wise Package Studio 42
password, Wise Package Studio 42
user account, server-side service 143
user group
See group
user security
See security
user-defined test case 172

validation
Also see Package Validation
error 147
of installation 146
terminal server compatibility 157
test descriptions 156
Validation directory 26
validation module
about 148
adding 148
customizing 148
predefined 156
selecting 146
selecting rules 149
validation rule
about 152
action 154
calling .DLL 151
calling .EXE 151
calling custom action 150
calling VBScript 151
calling .DLL 151
calling .EXE 151
calling VBScript 151
condition 154
creating 150, 153
editing predefined 155
selecting 149
variables
about 75
adding to help text 71
in Project Directory field 55
in Vendor Package field 56
list 75
VBScript file
calling with validation rule 151
vendor
name 56
package 56
VendorPackage variable 75
Verify Installation, test case 177
Virtual Capture
about 237
capture method 225
limitations 238
virtual memory page file, for SOE Snapshot 243
Virtual OS
about 237
creating 239
creation utility 239
selecting 226
Virtual OS directory 26
Virtual OS file
defined 32
Virtual Package Editor
about 88
command-line options 285
virtual software archive 32
virtual software layer
capturing in 219
running Test Expert 164
virtual software package 30
virtual software project 32
VSA file 32
W
wamdb.idx
about 27
Web application
adding as tool 68
allow connection attempts 52
connecting 79
override URL 69, 79
URL disabled 79
Web Capture Conversion 141
wildcard, excluding with 213
Windows 2000, validation checks 156
Windows 2003
with Preflight 265
Windows Application.msi 36
Windows Installer
administrative installation 257
advantages 15
capturing installation to 219
database 31
developer documentation 34
default 34
description requirements 156, 157
package, defined 31
project, defined 32
Windows Installer Editor
about 88
command-line options 286
not available 78
Windows Installer SDK Help 34
Windows NT
authentication 39
importing security group 42
Windows Vista
logo requirements 157
Windows XP, isolation 93
WinINSTALL package, converting 115
WinSxS directory 93
Wise package definition file
See package definition
Wise Services database
about 23
Wise share point
See share point directory
Wise Software Repository
about 23
active 23
adding files from 144, 206, 232
connecting 53
default, changing 53
multiples 23
Wise Software Repository database
See Software Manager database
Wise Task Manager
about 141
using 142
Wise Task Manager task 141
Wise Users group 41
Wise Web Capture
decrypting output 141
using 245
WiseComCapture.exe 207
WisePSSC.ini
about 202
basic exclusion list 247
using 205
WiseScript
capturing installation to 219
converting to Windows Installer 116
editing 89
file, defined 32
WiseScript Editor
about 88
command-line options 288
not available 78
WiseScript Package Editor
about 89
command-line options 288
not available 78
WiseVistaIce.cub 157
WLC file 49
Word, editing help text 71
WordPad, editing help text 71
Workbench
about 19
defined 13
hiding tools 68
interface 19
Projects tab 19, 76
resizing pane 22
setting up 38
Tools tab 21, 78
Workbench configuration, process 59
Workbench database
defined 23
external 60
global 60
multiples 60
setting permissions 46
Workbench directory 26
Workbench reports 85
Workbench tool
See tool
workbench tool, task type 63
WorkbenchDSN variable 75
WOS file
about 237
Also see Virtual OS
creating 239
defined 32
WPF file
creating 123
defined 32
WPR file 65
WPS Administrator group 41
WSE file
converting to Windows
Installer 116
defined 32
importing 116
WSI file
defined 32
editing 88
WSM file
Also see merge module
defined 32
WTE file 163
WVP file 32

X
XPlogo.cub 156
Z
ZAP file, copying 255