WinINSTALL 9.0
MSI Packager Professional

Administrators Guide

Disclaimer

The information contained in this publication is subject to change without notice. Attachmate Corporation makes no warranty of any kind with regard to this manual, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Attachmate Corporation shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this manual.

Copyright

TABLE OF CONTENTS

Section 1: Introduction 11

Chapter 1: Welcome to WinINSTALL 13

Recommended WinINSTALL Workflow 14
WinINSTALL Product Family 14
What's New in WinINSTALL 14
Remote Control 15
Two-Phase Distribution 15
New Patch Management Feature 16
Job Creation Wizard 16
IPC File Transfer 16
PXE Client Reset Enhancements 17
Enhanced Replication Checkpoint Restart 17

Chapter 2: Components 19

Components in the WinINSTALL Architecture 19
WinINSTALL Console 19
WinINSTALL Share 19
WinINSTALL Database 19
Reference Machine 20

Chapter 3: Navigating the WinINSTALL Console 23

Console Layout 23
Tree Pane 24
List Pane 25
Data Pane 26
Shortcuts Bar 27
Common Console Operations 28
How to view or Modify software distribution packages or lists 28
How to Create, edit or view conflict assessments 28
How to view the Console Log 28
How to Create, edit, or view reports 29
Customizing the Console 29

How to Start the Console where the last session Ended 29
To show/hide the toolbar: 29
How to show/hide the status bar 29
How to Customize the Shortcuts Bar 30
Setting Console Security 31
How to set Console Security 32
Administrative access rights: 32
Access rights for conflict assessment: 33
Access rights for packaging: 33

Section 2: Lists and Packages 35

Chapter 4: Lists and Packages 37
The Software Distribution Lists Node 37
Status Tab 38
Schedule Tab 41
Lists 43
How to Perform Common List File Operations 43
Windows Installer Packages (.MSI Files) 47
Windows Installer Package Data and Settings 47
How to Perform Common Package Operations 48
Build a Package 48
Add Packages 48
Edit a Package 49
Search and Replace Strings in a Package 50

Chapter 5: Building Packages 51
Using Discover to Build Packages Automatically 51
Launching Discover 51
Discover Reference Machine 52
Archived Before Snapshots 53
Basic Before Snapshots 55
Discover Advanced Options 55

Using Discover to Build a Windows Installer Package or Merge Module 57
Using Discover to Build a Windows Installer Transform 63
Embedding Transforms 71
Applying a Transform 72
Building Packages Manually 72
Build a Windows Installer Package Manually 73
Build a Windows installer Merge module Manually 73
Build a Windows Installer Transform Manually 74
Building Windows Installer Patches 75
The WinINSTALL Patch Wizard 75

Chapter 6: Package Validation and Repair 79

Validating a Windows Installer Package 80
How to validate a Windows Installer package 80
Repairing a Validated Windows Installer Package 81
How To repair a validated package 81

Chapter 7: Conflict Assessment 83

How to Generate a Conflict Assessment Baseline 84
The Conflict Assessment Wizard 84
Welcome 85
Baseline Selection 85
Package Selection 85
Confirmation 85
Completion 85
How to Edit a Conflict Assessment 86
How to Delete a Conflict Assessment 86
How to Run a Conflict Assessment 86
Viewing Conflict Assessment Results 87
How to view the results of a conflict assessment In the Console: 87
How to view the results of a conflict assessment in a report: 87
Conflict Assessment Categories 87
Main conflict categories 88
File conflict subcategories and messages 88
Registry conflict subcategories and messages 88

Shortcut conflict subcategories and messages 89
INI File conflict subcategories and messages 89

Chapter 8: Compression 91
MSI Package Compression 91
MSI File Compression Settings 91
Digital Signatures 92
Applying a Digital Signature to a Windows Installer Package 92
Signing an External Cabinet File 93

Chapter 9: Variables and Properties 95
How Variables Work 95
SYSTEM PROPERTIES 96
System Properties used in Windows Installer Packages 96
Adding system properties to an MSI package 98
Prompting the User for a Property Value 99

Chapter 10: Package Conditions 101
Windows Installer Launch Conditions 101
Package Launch Conditions 101
Feature Conditions 102

Section 3: Package Editing 105

Chapter 11: General Package Options 107
Summary Tab (Windows Installer package) 107
ARP Icon 107
Show Product in ARP 107
Show Change Button 107
Show Remove Button 107
Support Information 108
MSI Schema 109
Version 109

Product Code 109
Package Code 109
Product Language 110
Product ID 110
Manufacturer 110
Manufacturer's URL 110
Comments 110
Update package code 110
General Information for Installing a Windows Installer Feature 111
Summary Tab (Windows Installer Feature) 112
Conditions Tab (Windows Installer Feature) 113
Advanced Tab (Windows Installer Feature) 114
General Information for Installing a Windows Installer Component 116
Summary Tab (Windows Installer Component) 116
Reserve Cost Tab (Windows Installer Component) 118
Advanced Tab (Windows Installer Component) 119

Chapter 12: Additional MSI Package Options 121

Directory Tab 122
To manage directories used in a Windows Installer package: 122
Install Modes Tab 122
To see or enter Install Mode information for a package: 122
Advanced Tab 123
Components Sub-tab 124
Launch Condition Sub-tab 125
Sequence Sub-tab 125
The Custom Action Wizard 132
Source Type 133
Source Storage 133
Target Panels 133
Final Details 135
Completing the WinINSTALL Custom Action Wizard 136
MSI Custom Action Example 137
Defining the Custom Action: 137

Specifying the Custom Action Sequence: 138
The MSI Table Editor 140
How to Change Values in the MSI Table Editor 140
How to Save or Discard Changes in the MSI Table Editor 141

Chapter 13: Files 143
Windows Installer Package File Operations 143
Add Files 143
Remove Files 145
Move Files 145
Duplicate Files 146
Create Folders 147
ODBC Information 148
Search and Replace Feature 151

Chapter 14: Shortcuts 153
Adding or Changing Shortcuts with a Windows Installer (MSI) Package 153
Add Tab 153
Add/Edit Shortcut Dialog 154

Chapter 15: Registry 157
Registry Files 157
Registry Changes in a Windows Installer Package 158
Add Tab 158
Remove Tab 159

Chapter 16: System Services 163
Adding or Changing Services with a Windows Installer (MSI) Package 163
Adding, Editing, or Deleting System Services 163
Controlling System Services 166

Chapter 17: Advertising 169
How Advertising Works 169

Advertising Packages for Distribution 170
How to Advertise a Windows Installer package 170
Managing Programmatic Entry Points in a Windows Installer (MSI) Package or Merge Module 171

Chapter 18: File Edits 179

INI File Edits 179
Text File Edits 180
Editing Files with a Windows Installer (MSI) Package 180
INI Files Tab 180
ASCII Files Tab 182
Environment Tab 185

Chapter 19: Merge Modules 187

Merge Module Cache 187
Adding New Merge Module Folders to the Cache 188
Updating the Cache 188
Editing Merge Modules 188
Summary Tab 189
Advanced Tab 190
Merge Module Tab 191
Directory Tab 193

Section 4: Additional Operations 195

Chapter 20: Reports 197

How to Add a Report to the Console 198
How to Modify Report Properties 198
How to Delete a Report 198
How to Run a Report 199
How to Print a Report 199
How to Launch Crystal Designer from the Console 200

Chapter 21: Miscellaneous 201
WinINSTALL Logging 201
Console logging 201
WinINSTALL Utility Programs 203
Creating a Baseline for Conflict Assessment (WIBaselineGen.exe) 204
Script to Add a Database User (dbAddUser.cmd) 204

Index 205

Section 1

INTRODUCTION

CHAPTER 1: WELCOME TO WINSTALL

CHAPTER 2: COMPONENTS

CHAPTER 3: NAVIGATING THE WINSTALL CONSOLE

1

WELCOME TO WINSTALL

WinINSTALL is designed to help you meet all of your network desktop management needs - quickly, easily, and efficiently. To help you most quickly get started using WinINSTALL's wide range of capabilities, the documentation is divided into two separate guides: an *Installation Guide* and an *Administrators Guide* (this manual).

The *Installation Guide* presents installation requirements for each WinINSTALL component, plus instructions on how to install and configure each component, including basic instructions on agent deployment.

This *Administrators Guide* first introduces you to the new features in this exciting new release, presents the product architecture and workflow, helps you to familiarize yourself with the WinINSTALL Console, and then provides detailed instructions for using each of WinINSTALL's powerful features, including advanced operations.

The WinINSTALL workflow follows a logical, real-world sequence. First, you install the WinINSTALL Console and Share and set up your WinINSTALL database. Next, you configure your WinINSTALL environment. . Then, you are ready to use WInINSTALL's powerful array of integrated features..

- From the WinINSTALL console, you can take direct remote control of any managed workstation or server, either through Windows Remote Desktop Connection or through a remote control tool of your choice.
-
- You can build and organize packages - Windows Installer packages, merge modules, patches, and transforms.
- You can validate and repair individual Windows Installer packages and perform conflict assessments on groups of packages that will be installed together to pinpoint potential problems before they occur.

The WinINSTALL Console is your point of control for all of WinINSTALL's capabilities. This flexible tool has been designed to make it easy to use and easy to understand the array of powerful features it puts at your fingertips. New users and users new to this version should take a few minutes to familiarize themselves with the design and use of the Console. The chapter entitled *Navigating the WinINSTALL Console* provides a basic explanation of the operation of the WinINSTALL Console.

RECOMMENDED WININSTALL WORKFLOW

Once you have installed the product, configured your environment, and explored how to use the Console, you are ready to manage your computing environment. Using the recommended workflow, you will receive the maximum benefit from this powerful product.

- Build packages.
- Organize packages.
- Edit packages.
- Validate and repair packages.
- Pinpoint and resolve potential conflicts before you install packages.

Each step in the workflow is described in detail in later chapters of this manual.

WININSTALL PRODUCT FAMILY

The WinINSTALL® family of products was created for you - the network administrators who must install, manage, and maintain software installation in enterprise networks.

The WinINSTALL family of products includes the following:

- WinINSTALL Limited Edition (LE) MSI Packager
- WinINSTALL MSI Packager Professional Edition
- WinINSTALL Software Distribution Suite (SDS)
- WinINSTALL Desktop Management Suite (DMS)
- WinINSTALL Desktop Availability Suite (DAS)

Attachmate's WinINSTALL product line has been an industry leader in software distribution and management for over a decade. Our innovative technology, ease-of-use, and responsiveness to the needs of our customers have made us the front-runner in the desktop availability solutions market and the number one choice of network administrators in organizations of every size.

WHAT'S NEW IN WININSTALL

The WinINSTALL 9.0 family of products includes many new and improved features designed to make your job as a network administrator easier. The major new features and enhancements are listed below.

REMOTE CONTROL

WinINSTALL now allows a console user to remotely control any client machine directly from the Machines list through either of two methods:

- Windows Remote Desktop Connection (RDC).
- The remote control product of your choice.

See Remote Control, below, for more information on the new Remote Control feature.

TWO-PHASE DISTRIBUTION

WinINSTALL now offers a new option for delivering application packages to managed workstations. The new Two-Phase Distribution option delivers the package to the workstation in phase one, and then, in phase two, installs that package from the local drive. This feature provides a number of new capabilities to WinINSTALL's industry leading software distribution capability:

- Local execution of the WinINSTALL Installer.
- Bandwidth throttling during package delivery.
- Multicast package delivery option.
- Checkpoint restart (sub-file level) during package delivery.
- Package delivery without requiring access to a network share.
- Decoupling of package delivery from package installation.

This combination of new capabilities makes Two-Phase Distribution an ideal mechanism for delivering large packages to remote or poorly connected users.

***TIP:** This combination of new capabilities makes Two-Phase Distribution an ideal mechanism for delivering large packages to remote or poorly connected users.*

***NOTE:** Two-Phase Distribution requires certain firewall settings to operate. These settings are configurable.*

See the Two-Phase Distribution Jobs vs. Software Distribution Jobs section of the Package Distribution chapter for more information on Two-Phase Distribution.

NEW PATCH MANAGEMENT FEATURE

WinINSTALL now provides a comprehensive patch management feature with a number of new capabilities:

- Ability to schedule vulnerability scans as well as remediation (patch application) jobs.
- Ability to view all managed machines with a selected vulnerability.
- Ability to apply patches (remediations) to all vulnerable machines.
- Remediation jobs use the new Two-Phase Distribution feature, enabling all the associated advantages and options of that feature.
- Vulnerability scans and remediation jobs employ the Microsoft Windows Update Agent for maximum reliability and minimum system reboots.
- The vulnerability database and remediation patches can be automatically downloaded to WinINSTALL servers, enabling all vulnerability scans and remediation operations to be carried out with no required workstation internet access.
- Legacy patch delivery through WinINSTALL .nai packages remains an available option.

See the Patch Management chapter for full information on WinINSTALL Patch Management.

JOB CREATION WIZARD

WinINSTALL now provides a Wizard for job creation. This Wizard is a new, unified facility for creating both scheduled and immediate jobs, making running tasks on managed workstations easier and more intuitive than ever. See the Creating WinINSTALL Jobs chapter for full information on the *WinINSTALL Job Creation Wizard*.

IPC FILE TRANSFER

WinINSTALL has long used IPC (Inter-Process Communication) for direct interaction between the WinINSTALL console and managed machines. Now this facility is used optionally or automatically in several additional WinINSTALL operations:

- IPC file transfer is available as an optional means of transferring configuration and transaction files between the WinINSTALL agent on each managed machine and the WinINSTALL server. This new option, configured on the WinINSTALL Agent Settings Advanced Tab, allows these operations to make more efficient use of network bandwidth and strengthens network security.
- The new Two-Phase Distribution feature automatically uses IPC file transfer to move packages to the local drive of the target machine before installation.
- The new Patch Management facility also uses IPC file transfer to move patches to the local drive of each target machine before the patches are applied.

PXE CLIENT RESET ENHANCEMENTS

The PXE-based Client Reset feature of the WinINSTALL Desktop Availability Suite installs Windows operating systems directly from network servers, without loading MS-DOS or running MS-DOS applications. In addition, administrators can use the built-in integration of WinINSTALL software distribution and personality transfer to optionally include in the reset process installation of any number of application packages and/or Microsoft patches, and the restoration of user data and settings as well.

This release introduces a number of significant enhancements to this major WinINSTALL feature:

- PXE Reset queues.
- Improved automatic OS language detection.
- Automatic OS detection for drivers.
- SCSI/SATA driver support.
- Regional language setting (Locale) support.
- The UI now enables the hiding of unused languages.
- Support for disabling the administrator account on PXE reset clients.

ENHANCED REPLICATION CHECKPOINT RESTART

The server-to-server replication feature included in the WinINSTALL Desktop Management Suite and WinINSTALL Desktop Availability Suite has always provided a file-level checkpoint restart capability for interrupted replication jobs. Now this checkpoint restart capability has been extended to sub-file level, vastly increasing the efficiency of this feature. For example, if a replication session is interrupted 90% of the way through the transmission of a large file, WinINSTALL will now transfer only the remaining 10% of the file, rather than the entire file, when the session is reestablished.

NOTE: *This enhanced checkpoint restart feature is automatically used in all Two-Phase Distribution jobs as well as in server to server replication.*

2

COMPONENTS

WinINSTALL is composed of the following separate components that work together within the WinINSTALL architecture to provide powerful, yet flexible desktop management capabilities.

- Console
- Share
- Database
- Reference machine

COMPONENTS IN THE WININSTALL ARCHITECTURE

Each of the WinINSTALL components is described below.

WININSTALL CONSOLE

The WinINSTALL Console is the graphical user interface (GUI) used to administer the entire WinINSTALL environment. The Console lets you view information in the WinINSTALL database to which the Console is connected. The Console can be installed on any machine. The Console machine must have access to the WinINSTALL database and have any required database client software installed. There can be one or multiple Consoles. Each Console can be configured to allow multiple administrators the ability to access and control specific WinINSTALL desktop management capabilities and to deny access to other capabilities.

WININSTALL SHARE

The WinINSTALL share serves multiple purposes. First of all, it holds the files used by the Console, such as configuration files, executables, help files, report files, and software packages.

The WinINSTALL share is the repository for software distribution packages and Microsoft patches. It is here where the Console creates and edits these packages..

WININSTALL DATABASE

The WinINSTALL database contains information entered through the WinINSTALL Console for package management, logging, and conflict assessment. The data in the

database can be viewed and manipulated from the Console, the graphical user interface for WinINSTALL.

DATABASE SCHEMA

A complete listing of all tables in the WinINSTALL database is included with the product as a separate file, *DBSchema.htm*, located in the WinINSTALL share in the *\Bin\Help* folder. You can use this information with third-party database mining tools and to create custom reports, if you have a copy of Crystal Designer.

The *DBSchema.htm* file documents all tables in the WinINSTALL database, presenting information on each database table, including table name, column name, data type, column size, and precision (where applicable) for both SQL Server and Oracle, plus whether the field is required or not.

For character data types, the length shown is the length in characters. For other data types, the specified length is the number of bytes occupied by the value within the table. Some types, such as text and image, occupy space in a special place in the database, because they are often too large for a normal row. These types have 16 bytes of overhead in the row. Similar Oracle types, (LONG and LONG RAW), do not have a length specified, for the same reason.

The tables are grouped within the list according to the WinINSTALL components which make primary use of them. Note that, despite their grouping in the list, many of these tables are used by multiple WinINSTALL features and components, and often for purposes which are not obvious.

REFERENCE MACHINE

A reference machine is a special machine used solely for the purpose of building packages and creating transforms. No WinINSTALL components are installed on a reference machine.

Technically, a reference machine should be a clean machine, with only an operating system and any necessary service packs installed. Because a reference machine does not need to be large and expensive, some administrators have two or more. In this way, they can create a package on one machine while returning a previously-used machine to its clean state. This practice reduces downtime - and the temptation to cut corners by using a dirty machine.

Today, a number of commercial products are available to simplify the process of returning a reference machine to its clean state. For instance, virtual machine software allows you to save a virtual machine in a clean state. After you build a package on a virtual machine, you can discard all changes and return the machine to its original clean state during reboot. Virtual machines can save a great deal of time because you need to install each operating system used in your network only once. You can create separate virtual machines for each operating system and then reuse them over and over again. This is particularly useful

because each package you build must be created on a machine with the same operating system as the network client on which it will be installed.

3

NAVIGATING THE WININSTALL CONSOLE

The WinINSTALL console is the graphical user interface (GUI) tool used to administer the entire WinINSTALL environment. The data displayed in the console is the data in the database selected in the *Default Settings Wizard* or on the *Console Options* dialog.

If you change from one database to another, you change the data that you see in the console.

***WARNING:** Console database connection information is stored on the share and used by any console accessing that share. So if you change the database you are viewing in the console, you are changing the database being viewed by any console connected to that share.*

Once you understand the console layout, you can customize it and add security to it. The minimum video resolution for the WinINSTALL Console is 1024 x 768.

CONSOLE LAYOUT

The Console is divided into three panes - tree pane, list pane, and data pane - with a shortcut bar on the far left and a context-sensitive menu bar and toolbar at the top.

Tree Pane

The tree pane is the top left pane of the Console. The nodes in the tree pane allow you to access the major functional areas of the product. Clicking a tree node can reveal sub-nodes that represent nested functionality within that area. The node selected in the tree pane determines the appearance of the data pane and, for some nodes, the appearance of the list pane as well.

List Pane

The list pane is the bottom left pane of the Console. Context-sensitive information displays in this pane when certain sub-nodes are selected in the tree pane. The entry selected in the list pane then determines the appearance of the data pane.

Data Pane

The data pane is the right pane of the Console. Detailed context-sensitive information displays in a tabbed format in this pane. The entry selected in either the tree pane or the combined entries selected in the tree pane and the list pane determine the appearance of the data pane. The data pane is where you view and edit information.

Shortcuts Bar

The shortcuts bar displays on the far left side of the Console. With a simple menu click, you can hide this bar or make it visible again. You can easily customize this bar to reflect the needs of your environment.

Menu bar and Toolbar

At the top of the Console is a context-sensitive menu bar and toolbar. The menu categories, menu items, and toolbar icons change depending on the selection made in the tree pane of the console.

TREE PANE

The tree pane is the top left pane in the console. The node that is selected in the tree pane determines the categories that display in the list pane (below the tree pane) and the context sensitive information that displays in the data pane (on the right of the tree pane). Navigating the tree pane is similar to navigating Microsoft's Windows™ Explorer. To expand a node, click the plus sign beside the node. To collapse an expanded node, click the minus sign beside the node.

The top-level nodes in the tree pane represent major functional areas of the product. Clicking the sub-nodes below reveal nested functionality within that area. For example, under the *Software Distribution* node, lists can be expanded to reveal packages and nested child lists, packages can be expanded to reveal the features they contain, and features can be expanded to reveal their components or sub-features. The nodes in the tree pane are outlined below.

START PAGE

This node provides a graphical overview of the features in the WinINSTALL family of products and access to a *Getting Started Guide* that helps the administrator learn to use the product quickly, with direct links to product functionality.

SOFTWARE DISTRIBUTION

This node allows you to create and modify software distribution packages and merge modules (.MSM files).

- Software Distribution Lists

Please see the Lists and Packages chapter for additional information on software distribution lists and packages.

- Merge Module Folders

The Merge Module Folders node can be configured to hold the Merge Module Cache (WinINSTALL provides a number of Microsoft-supplied merge modules), which is automatically consulted for matching features and components when building a new package. Please see the Merge Modules chapter for additional information.

CONFLICT ASSESSMENT

This node allows you to check for conflicts between packages and specific operating system environments or between groups of packages. See the Conflict Assessment chapter for additional information on this feature.

CONSOLE LOG

This node allows you to view user activity that the Console has logged to the WinINSTALL database and the Windows Event Log. (The Console must have been configured to write information to one or both of these logs, using the *Console Options* dialog. Please see the WinINSTALL Logging section of the Miscellaneous chapter for details on the WinINSTALL console log.)

REPORTS

This node provides access to the full array of prepackaged reports and to Crystal Designer, if you have a licensed copy of the Crystal reports software installed on the Console machine. For more information, see the Reports chapter.

LIST PANE

The list pane is the bottom left pane of the Console. Categories of information display in this pane when certain sub-nodes are selected in the tree pane. Each category in the list pane causes context-sensitive detailed information to display on the right side of the screen, in the data pane.

SOFTWARE DISTRIBUTION

When a Windows Installer package, feature, or component is highlighted under the *Software Distribution* node in the tree pane, the categories below display in the list pane.

- General
- Files
- Shortcuts
- Registry
- Services
- Edits

- Advertising

You can also expand a Windows Installer package to the feature or component level. Selecting a feature or component displays a different list of categories from a package level selection.

DATA PANE

The data pane is the right pane of the Console. Detailed context-sensitive information displays in a tabbed format, with the appearance and content determined by the node that is selected in the tree pane and, in many cases, the category that is selected in the list pane as well. In the data pane, you can view and modify detailed information about the selected node or sub-node.

EDITING BUTTONS

A common feature of the data pane user interface is a set of buttons, each marked with a different icon. Typically, the user interface will present a list of entities (such as files to be copied as part of a package) with these buttons arranged along the top.

- Buttons with these icons enable you to add, modify, delete and move items in a list.

Each button is always used in the same fashion, regardless of the nature of the entities in the list. Hovering the mouse over an enabled button will produce a tool tip that suggests the function of that button. The function of each button is explained below.

- *Add New* Icon

Clicking this button will create a new entity. When the entity is complete, it will be added to the list, where it can be acted upon by means of the buttons marked with Edit, Delete, Up Arrow and Down Arrow icons.

- *Edit* or *Properties* Icon

Clicking a button marked with the Edit Icon will present the properties of the selected entity for editing. When the editing process is complete, the entity will be returned to the list.

- *Delete* Icon

Clicking a button marked with the Delete Icon will remove the selected entity from the list.

- *Up Arrow* Icon

Clicking a button with this icon will move the selected entity up one row in the list.

- *Down Arrow* Icon

Clicking a button with this icon will move the selected entity down one row in the list.

- *Ellipsis* Icon

Clicking a button with this icon will present a screen with additional parameters and settings.

SHORTCUTS BAR

The shortcuts bar displays on the far left side of the console. The shortcuts bar is convenient for accessing items or screens that you use most frequently or for items that appear at the bottom of long lists or hierarchies. You can customize this bar to reflect the needs of your environment. You can choose to hide this bar or keep it visible.

***TIP:** Just drag and drop items from any major area of the Console to the shortcuts bar. For example, you can bookmark specific tabbed screens, field names, and group box names*

COMMON CONSOLE OPERATIONS

HOW TO VIEW OR MODIFY SOFTWARE DISTRIBUTION PACKAGES OR LISTS

1. **Highlight the *Software Distribution* node in the tree pane.**
2. **Highlight the *Software Distribution Lists* node in the tree pane.**
3. **Highlight a specific software distribution list in the tree pane.**
4. **View or modify a software distribution list.**
5. **Highlight a specific package in the tree pane.**
6. **View or modify a Windows Installer (MSI) package, feature, or component.**
7. **View or modify a WinINSTALL (NAI) package.**

HOW TO CREATE, EDIT OR VIEW CONFLICT ASSESSMENTS

1. **Highlight the *Conflict Assessment* node in the tree pane.**
2. **Create, view or modify conflict assessments.**

HOW TO VIEW THE CONSOLE LOG

1. **Highlight the *Console Log* node in the tree pane.**
2. **View logged data on either the Database or Windows Event Log tab.**

TIP: *In both the Database and Windows Event Logs, you can use the* Edit *menu or the right-click context menu in the log to copy all or selected entries to the clipboard for pasting into other applications, or to export all or selected entries to a tab-delimited text file for processing by another application. The Export and Export All options will prompt you for an output path and filename.*

HOW TO CREATE, EDIT, OR VIEW REPORTS

1. **Expand the *Reports* node in the tree pane.**
2. **Highlight a report category in the tree pane.**
3. **Run, create or modify a report.**

CUSTOMIZING THE CONSOLE

You can customize your WinINSTALL Console in several ways to enable you to work according to your own preferences.

HOW TO START THE CONSOLE WHERE THE LAST SESSION ENDED

1. **Click *View* on the menu bar.**
2. **Choose *Startup at Last Location*. (It will now be checked, causing future executions of the console to start where the last session ended.)**

TO SHOW/HIDE THE TOOLBAR:

1. **Click *View* on the menu bar.**
2. **Choose *Bars*.**
3. **Click *Tool Bar*. If *Tool Bar* is checked, the toolbar will be visible; otherwise, it will be hidden.**

HOW TO SHOW/HIDE THE STATUS BAR

1. **Click *View* on the menu bar.**
2. **Choose *Bars*.**
3. **Click *Status Bar*. If *Status Bar* is checked, the status bar will be visible; otherwise, it will be hidden.**

HOW TO CUSTOMIZE THE SHORTCUTS BAR

SHOW/HIDE THE SHORTCUTS BAR

1. Click ***View*** on the menu bar.
2. Choose ***Bars***.
3. Click ***Shortcuts Bar***. If ***Shortcuts Bar*** is checked, the shortcuts bar will be visible; otherwise, ti will be hidden.

ADD A NEW SHORTCUT BAR

1. Right-click in the Shortcuts Bar area.
2. Select ***Add Bars***.
3. The ***Add Shortcut Bars*** dialog displays.
4. Enter the name of the new shortcuts bar and click ***Add***.
5. The name of the new shortcut bar now displays at the top of the Shortcuts Bar area.

REMOVE AN EXISTING SHORTCUT BAR

1. Right-click in the Shortcuts Bar area.
2. Select ***Remove Bars***.
3. The ***Remove Shortcut Bars*** dialog displays.
4. From the drop-down list, select the shortcut bar that you want to remove and click ***Remove***.
5. The name of the deleted shortcut bar no longer displays at the top or bottom of the Shortcuts Bar area.

MOVE AN EXISTING SHORTCUT BAR

1. Right-click in the Shortcuts Bar area.
2. Select ***Move Bars***.
3. The ***Move Shortcut Bars*** dialog displays.

4 **From the drop-down list, select the shortcut bar that you want to move and click *Move*.**

5 **The name of the shortcut bar moves to the bottom of the list at the top of the Shortcuts Bar area and the shortcut bar opens.**

EDIT THE NAME OF A SHORTCUT BAR

1 **Right-click in the Shortcuts Bar area.**

2 **Select *Edit Bar Names*.**

3 **The *Edit Shortcut Bar Names* dialog displays.**

4 **From the drop-down list, select the shortcut bar that you want to rename and click *Rename*.**

5 **The shortcut bar now displays with its new name on the Shortcuts Bar area.**

RENAME A SHORTCUT ON A SHORTCUT BAR

1 **Right-click a shortcut icon or name on a shortcut bar.**

2 **Select *Edit Shortcut Name*.**

3 **Edit the shortcut name.**

4 **Press *Enter*.**

5 **The shortcut now displays with the modified name.**

DELETE A SHORTCUT FROM A SHORTCUT BAR

1 **Right-click a shortcut icon or name on a shortcut bar.**

2 **Select *Delete Shortcut*.**

3 **The shortcut icon disappears from the shortcut bar.**

SETTING CONSOLE SECURITY

You can set the access rights of specific users and groups for each area of functionality available from the WinINSTALL Console. These security settings cover only the activity of the Console user and do not apply to other parts of WinINSTALL, such as Discover.

HOW TO SET CONSOLE SECURITY

1. **Select *Security* from the *View* menu.**
2. **The *Console Security* dialog will appear, listing high-level Console features on the left and providing a *Configure* button on the right. Select a feature on the left and click the *Configure* button.**
3. **A standard Windows security dialog for will be displayed, allowing you to set privileges for the selected feature.**

For each Console feature, the logged-on user of the console must have the proper access right(s) assigned in order to perform the specific activities of that feature. Access rights may be assigned to users or groups. The logged-on user of the console will not only have the access rights granted specifically to him or her, but will also inherit any access rights given to groups of which (s)he is a member.

Menus, toolbars, and console tree nodes are not directly affected by the access rights of the user. In other words, menu commands are not enabled or disabled based on security. Instead, when a user attempts to perform an action for which (s)he does not have the proper rights, an error message appears indicating that fact.

ADMINISTRATIVE ACCESS RIGHTS:

Full Administrative access rights include the ability to perform the following tasks:

- Set Console Options (database, share, logging, etc.).
- Set Security Permissions.
- Add/Edit/Delete top-level (directly under the Software Distribution Lists node in the tree view) Lists.
- Add/Edit/Delete Reports.
- Access all Console functions (overrides any other access rights assigned or not assigned for Console features – i.e., the other rights described below).

Denying Administrative access rights will produce the following behavior in the Console:

- *Console Options* dialog will not appear.
- *Console Security* dialog will not appear.
- Cannot edit/delete machines.
- Cannot add/edit/delete top-level lists.

- Cannot add/edit/delete reports.

***WARNING:** Before denying Administrative access rights, be sure you have specifically granted them to yourself. Otherwise, you may inadvertently prevent yourself from performing administrative actions, including making further Console security changes.*

ACCESS RIGHTS FOR CONFLICT ASSESSMENT:

Allowing full access rights for Conflict Assessment will permit the following actions:

- Launch *Conflict Assessment Wizard*.
- Add/edit/delete Conflict Assessments.

Allowing access rights to view Conflict Assessments will permit the user to view Conflict Assessments only. Access to the *Conflict Assessment Wizard* and to adding/editing or deleting Conflict Assessments will be denied.

Denying all Conflict Assessment access rights will produce the following behavior:

- The *Conflict Assessment Wizard* will not appear.
- The user will be unable to view, add, edit, or delete Conflict Assessments.

ACCESS RIGHTS FOR PACKAGING:

Full access rights for Packaging enable the user to add/edit/delete lists, packages, and merge modules.

View only access rights to Packaging allows the user to view packages and merge modules, but not to modify them in any way.

Denying access rights for Packaging will produce an “Access Denied” message when the user selects the *Software Distribution* tree node.

Section 2

Lists and Packages

Chapter 4: Lists and Packages

Chapter 5: Building Packages

Chapter 6: Package Validation and Repair

Chapter 7: Conflict Assessment

Chapter 8: Compression

Chapter 9: Variables and Properties

Chapter 10: Package Conditions

4

LISTS AND PACKAGES

From the WinINSTALL Console, you can create and edit packages and organize them in containers called lists. (Lists are special WinINSTALL files with a .lst extension.) You can create packages manually or let the Discover Wizard do the work for you.

When you build a package manually in the console, it is automatically added to the console when you save it. When you use the *Discover Wizard* to build a package, you have two choices:

- If you tell Discover to add the package to a list that already appears in the console, the package is automatically added to the console.
- If you don't specify a list or if you tell Discover to add it to a list that does *not* appear in the console, you must later add the package to the console manually.

From the WinINSTALL Console, you can also create and edit Windows Installer merge modules, transforms, and patches. You create merge modules and transforms with the *Discover Wizard*. You create patches with the *Patch Wizard*.

In addition, you can add to the console existing packages (created outside of WinINSTALL), such as third-party Windows Installer packages, merge modules, transforms, or patches. All packages that are added to the console can be edited in the console, regardless of how they were originally created.

Packages and merge modules display in the tree pane in the console under the *Software Distribution* node.

- Packages display under the *Software Distribution node,* beneath the *Software Distribution Lists* node, within any lists or sub-lists you have added.
- Merge modules display under the *Software Distribution* node, beneath the *Merge Module Folders* node, within any merge module folders you have added (none are present by default).

THE SOFTWARE DISTRIBUTION LISTS NODE

All packages and lists of packages managed by WinINSTALL are located beneath the *Software Distribution Lists* sub-node of the *Software Distribution* tree node.

If you click on the *Software Distribution Lists* node, the data pane will display two tabs: the Status Tab and the Schedule Tab. The *Status* and *Schedule* tabs recur throughout the package hierarchy, as follows.

First of all, the *Status* and *Schedule* tabs at the *Software Distribution Lists* level provide a comprehensive view of all software distribution jobs you have run or scheduled from the WinINSTALL console. The *Status* tab includes all jobs, both scheduled and immediate. The *Schedule* tab provides a view of scheduled software distribution jobs only.

At the list file level, a *Status* tab and a *Schedule* tab associated with each list file provide a view of this same information, filtered to include only those jobs involving the selected list file.

Finally, each package also provides these same two tabs, narrowing the focus to only those jobs involving the selected package.

STATUS TAB

Listed on the *Status* tab are all software distribution jobs associated with the *Software Distribution Lists* node, or with a specific list file or package, including jobs created as both immediate software distribution tasks and scheduled software jobs. In other words, the information on the Status tab concerns all software distribution jobs, both immediate and scheduled, but the jobs shown are filtered according to the context:

Software Distribution Lists node • All immediate and scheduled software distribution jobs

List file • All immediate and scheduled software distribution jobs involving the selected list file.

Package • All immediate and scheduled software distribution jobs involving the selected package.

For each listed job, the following information is provided on the tab:

Job Name • Assigned to the job when it was created on the *Job Settings* page of the *WinINSTALL Job Wizard* or the *Job Settings* tab of the *Edit Job* dialog.

Type • *Single phase* or *Two phase*; and *install*, *uninstall*, *reinstall*, or *command line*.

Date • The date and time when the job was created.

Status • One of the following status designations: *Scheduled*, *In progress*, *Succeeded*, *Partially succeeded*, or *Failed*.

You can perform the following actions on this tab:

- To retry a failed job, select the desired job and either click the *Retry* button, or right-click and select *Retry* from the context menu.
- To remove the status records for a scheduled distribution job from the database, select the desired job and click the *Purge* button.

***TIP:** When a scheduled job is purged, the job itself remains in the database, but the results are purged, leaving the job as though it has not yet been executed.*

- To delete an immediate job, select the desired job in the list and either click the *Delete* button or right-click and select *Delete* from the context menu.

***NOTE:** You can delete only immediate jobs on this tab. To delete a scheduled software distribution job, click the* Schedule *tab.*

- To view job details, either click the desired job and click the *Details* button or right-click and select *Details* from the context menu. The Distribution Job Status Details dialog will appear, displaying a collection of details on the status of the selected job.
- To refresh the status of the jobs in the list, click the *Refresh* button.
- To copy jobs in the list to the clipboard, select the desired jobs and either right-click and select *Copy* from the context menu or select *Copy from the Edit* menu. You can also click the *Edit* menu and select *Copy All* to copy the entire list to the clipboard.
- To export jobs in the list to a tab-delimited text file, select the desired jobs and either right-click and select *Export* from the context menu or select *Export* from the *Edit* menu. You can also click the *Edit* menu and select *Export All* to export the entire list to a tab-delimited text file.

DISTRIBUTION JOB STATUS DETAILS

If you select a job in the list on the *Status* tab and double-click, click the *Details* button, or right-click and select *Details* from the context menu, the *Distribution Job Status Details* dialog will appear.

This dialog is entirely read-only, and it is divided into the following four sections:

JOB DESCRIPTION

This section displays the following configuration details for the selected job. All of the displayed settings were specified in the original job creation through the WinINSTALL Job Wizard.

Job name
- The job name provided when the job was created.

Job type
- *Software distribution* or *Two phase distribution*.

Distribution type
- *Install*, *uninstall*, *reinstall*, or *command line*.

Package/list
- Path or file name of the package or list being distributed.

Suspended
- *Yes* if the job is currently suspended, *No* otherwise.

Canceled
- *Yes* if the job has been canceled, *No* otherwise.

Run type (Software distribution jobs only)
- Either *Scheduled* or *Run immediately.*

Download run type (Two phase distribution jobs only)
- Either *Scheduled* or *Run immediately.*

Install run type (Two phase distribution jobs only)
- Either *Scheduled* or *Run immediately after download*.

Download method (Two phase distribution jobs only)
- Refers to the server-to-workstation connection: *Automatic*, *Multicast*, *Unicast*, or *Unicast (workstation initiated)*.

Bandwidth throttle (Two phase distribution jobs only)
- If bandwidth throttling has been enabled for this job, the specified KB per second limit is displayed; otherwise, this field will show *None*.

Date created
- This field displays the date and time when the job was originally created.

JOB STATUS ON EACH TARGET MACHINE

This section displays a list of all target machines, with the following columns of information displayed for each machine in the list:

Machine
- The name of each target machine appears in the first column.

Date
- The second column displays the date and time when the record for each machine was last updated.

Download status • Status of the download phase for each machine (*Scheduled*, *In progress*, *Succeeded*, *Failed*) for Two phase distribution jobs, or - (blank) if the job is a Software distribution job.

Install status • Status of the installation for each machine (*Scheduled*, *In progress*, *Succeeded*, *Failed*).

SUMMARY STATISTICS

This section displays overall job statistics for the machine states listed below.

- Machines succeeded
- Machines failed
- Machines pending
- Machines running

Each statistic listed consists of two numbers (number in the specified state/total number of machines) followed by the percentage of machines in the specified state. For example, if 15 machines out of 100 target machines have succeeded, then the Machines succeeded number would display as follows: *15/100 (15%)*.

PACKAGE STATISTICS (SELECTED MACHINE)

This section displays the individual machine statistics for the list or package specified in the job.

- Packages found
- Packages installed
- Packages failing to install
- Packages not qualified or already installed

The first statistic, *Packages found*, is the simple number of specified packages the selected machine was able to find.

Each other statistic listed consists of two numbers (number in the specified state/total number of packages) followed by the percentage of packages in the specified state. For example, if 5 packages out of 10 included in the list file have succeeded, then the packages installed number would display as follows: *5/10 (50%)*.

SCHEDULE TAB

The information on the *Schedule* tab associated with the *Software Distribution Lists* node, or with a specific list file or package, is the same as that found on the *Schedule* tab in the data

pane of the console *Schedule* tree node--but filtered to display only scheduled software distribution jobs, and further filtered, like the Status Tab, according to the context:

Software Distribution Lists node	• All scheduled software distribution jobs
List file	• All scheduled software distribution jobs involving the selected list file.
Package	• All scheduled software distribution jobs involving the selected package.

The *Schedule* tab presents a list of all scheduled software distribution jobs and lets you add new scheduled tasks and modify, cancel, or delete existing scheduled jobs. The following information appears for each listed job:

Name	• The name of the job.
Task	• The type of job scheduled - in this case, Software Distribution or Two-Phase Distribution.
Target	• Whether the job is scheduled for specific machines or users.
Frequency	• How often the job will be run: Once, Daily, Weekly, Monthly, Yearly
Start	• The time at which the job is scheduled to start.
End	• The time at which the job is scheduled to end.

You can view, add, edit, suspend, resume, cancel, and delete the jobs listed on this tab:

- Click the *Add* button to launch the *WinINSTALL Job Wizard* to schedule a new job.
- To edit an existing job, either double-click the desired job or select it in the list and click the *Edit* button. The *Edit Job* dialog will appear, allowing you to edit the properties of the selected job.
- Click the *Suspend* button to suspend (pause) the execution of a scheduled job. The schedule will be suspended until you specify to resume it.
- Click the *Resume* button to resume execution of a scheduled job which has previously been suspended.
- Click the *Cancel* button to start the process of notifying the WinINSTALL agents on the appropriate machines that the job has been cancelled and should be removed from the machines. See below for more information on canceling and deleting scheduled jobs.
- Click the *Delete* button to permanently delete the selected job. Note that there are two steps in permanently removing a job that has been previously scheduled - first cancelling the job and then deleting the job. (The *Delete* button is not active until the state of the job

is *Canceled*. If you have selected multiple jobs, the *Delete* button is not active unless all of the selected jobs have been canceled.)

***NOTE:** The two-step process for deleting a scheduled job is to prevent scheduled jobs from being erroneously run after deletion as a result of timing issues between merging and publishing scheduling information to and from the database.*

LISTS

You organize packages in lists, which are special WinINSTALL files with a .lst extension. Lists can contain both packages and other lists, nested in a hierarchical tree structure. All lists are nested beneath the *Software Distribution Lists* node within the *Software Distribution* node in the tree pane.

You can create a new list and add it beneath *Software Distribution Lists*, or you can place an existing list beneath that node. You can also add a new or existing list to any nested child list. As you add new or existing lists, they display in the tree view of the console. When you add an existing list, you automatically add its contents as well (any packages or lists contained within that list). When you expand an existing list that you have added, the packages and lists that it contains display beneath the list in the console.

WinINSTALL contains a facility that automatically alphabetizes the contents of a list file and another facility that allows you to reorder the contents of a list file in whatever sequence you wish. Since adding a new package automatically places it at the end of a list, the ability to alphabetize or reorder the packages in a list provides a way for you to put new packages in a place where you can more easily find them.

HOW TO PERFORM COMMON LIST FILE OPERATIONS

CREATE A NEW LIST AND ADD IT TO THE CONSOLE

1. **Expand the *Software Distribution* node in the tree pane and highlight any list or sub-list beneath it.**
2. **Do one of the following:**
3. **Right-click the list and select *New* from the pop-up menu.**
4. **Choose *New* from the *File* menu on the menu bar.**
5. **Enter the required information on the *Create Application List* dialog and click *OK*.**

ADD AN EXISTING LIST TO THE CONSOLE

1. **Expand the *Software Distribution* node in the tree pane and highlight any list or sub-list beneath it. The list you are adding will appear within the selected list.**
2. **To add the list, do one of the following:**
   - **Click the list icon on the toolbar:** 
   - **Click the *Actions* menu on the menu bar and then choose *Add Existing --> Application List*.**
   - **Right-click the desired list and choose *Add Existing --> Application List* from the pop-up menu.**
3. **On the *Open* dialog, navigate to the list file you want to add and click OK.**

REMOVE A LIST FROM THE CONSOLE

1. **In the tree pane, expand the *Software Distribution* node.**
2. **Highlight the list or sub-list that you want to remove and do one of the following:**
   - **Right-click the list and choose *Delete* from the pop-up menu.**
   - **Choose *Delete* from the *File* menu on the menu bar.**
3. **You will then be asked to confirm the removal of the highlighted list. Click *OK* and the list file will no longer be displayed in the Console.**

ADD AN EXISTING PACKAGE TO THE CONSOLE

1. **In the tree pane, expand the *Software Distribution* node.**
2. **Highlight the list or sub-list to which you want to add a package and do one of the following:**
   - **Click the icon on the toolbar to add the package:**

      **(for Windows Installer package)**
   - **Right-click the list and choose *Add Existing --> Windows Installer Package* from the pop-up menu.**
   - **From the *Actions* menu on the menu bar, choose *Add Existing --> Windows Installer Package*.**
3. **On the *Open* dialog, navigate to the package that you want to add and click *OK*.**

REMOVE A PACKAGE FROM THE CONSOLE

1. **In the tree pane, expand the *Software Distribution* node and a nested list or lists, if necessary.**

2. **Highlight the package that you want to remove and do one of the following:**
3. **Right-click the package and choose *Delete* from the pop-up menu.**
4. **Choose *Delete* from the *File* menu on the menu bar.**
5. **You will be asked to confirm the removal of the highlighted package. Click *Yes* and the package will no longer be displayed in the list.**
6. **You will be prompted again, this time asking whether to delete the actual package file.**
   - **If you click *Yes*, the file is permanently deleted and cannot be added back to the Console.**
   - **If you click *No*, the package is no longer displayed in the list, but it still exists and can be added to any list as desired.**

ALPHABETIZE THE CONTENTS OF A LIST

1. **In the tree pane, expand the *Software Distribution* node.**
2. **Highlight the list or sub-list whose contents you want to alphabetize and do one of the following:**
   - **Right-click the list and choose *Alphabetize List* from the pop-up menu.**
   - **Choose *Alphabetize List* from the *Actions* menu.**

   The packages within the list will now appear in alphabetical order.

REORDER THE CONTENTS OF A LIST

1. **In the tree pane, expand the *Software Distribution* node.**
2. **Highlight the list or sub-list whose contents you want to reorder and do one of the following:**
   - **Right-click the list and choose *Reorder List* from the pop-up menu.**
   - **Choose *Reorder List* from the *Actions* menu.**
3. **On the *Reorder List* dialog, use the *Up* and *Down* arrow keys to manipulate the order of the packages in the list and click *OK*.**

   The packages within the list will now appear in the order you specified.

WinINSTALL uses the process of inheritance to set certain properties of list files and the objects contained by list files. These objects include WinINSTALL (NAI) files, Windows Installer (MSI) files, and other nested list files. A list file is a parent container, and the NAI and/or MSI files are child objects within that container. (Sub-lists within a parent list -

although they are children of the parent list - themselves become parent lists of the packages within them.)

The child objects - packages and sub-lists - inherit the settings of their parent list. For example, if you enable logging to the Windows Event Log for a list file, all of the packages within that list file will also have logging to the Windows Event Log enabled. You can, however, override the settings of a parent list by making local settings at the package or sub-list level.

Inheritance applies only when the installer processes a list file, and the inherited settings begin only at the level of the list file being processed.

When the installer processes a package or a sub-list directly, it has no knowledge of any parent list and therefore cannot apply any settings from a parent list. In other words, a list or a package referenced directly (i.e., not through a list) does not inherit anything from a parent, because it has no parent.

Local settings (i.e., settings at the package level) override inherited settings (i.e., settings at the containing list level), which in turn override WinINSTALL default settings.

The effective settings on a parent become the inherited settings for a child - when that child is processed while the installer is processing the parent.

WinINSTALL uses the following icons on the tabs in the data pane to provide a quick visual indication of the types of settings that are currently in place:


Default: The default value will be used for this setting when the package is processed.


Inherited: The value for this setting is inherited from a list in which the package is located. (Since lists can be nested, the actual value of this setting will be determined by inheritance rules.)


Local: The value for this setting has been set locally for this specific package.


Reset to Default: Clicking this icon resets the data to the program default setting. The value is saved as a local value.

Changing a value with one of these icons beside it will change the icon to its appropriate value.

WINDOWS INSTALLER PACKAGES (.MSI FILES)

Also called MSI packages, Windows Installer packages are application packages with the file extension .msi. MSI packages can be built by the *Discover Wizard*, created manually from the WinINSTALL Console, or created outside of WinINSTALL and added to the console. Regardless of how they are created, they can all be edited from within the WinINSTALL Console.

An MSI package is made up of one or more features, which are in turn made up of one or more components. Because the structure of an MSI package is very different from the structure of an NAI package, some of the editing methods differ for the two types of packages.

MSI packages are processed by the Windows Installer. When such a package is run from within WinINSTALL, the WinINSTALL installer is invoked first. The WinINSTALL installer processes certain settings to determine whether or not the package is qualified for the target machine. If no specified prerequisites are missing, it then launches the Windows Installer to process the actual installation of the package.

WINDOWS INSTALLER PACKAGE DATA AND SETTINGS

When you add or highlight a Windows Installer package, feature, or component in the tree pane, a list of applicable categories appears in the list pane. As you click each category, context-sensitive tabs and sub-tabs display in the data pane. On these tabs and sub-tabs, you can view, enter, modify or delete the detailed information that governs the actions performed when the package, feature, or component is installed.

In most cases, the appearance of the data tabs is the same, regardless of whether a component, feature or package is selected. However, the scope and amount of information available on the tab is different. If a component is selected, only information for that component is available. If a feature is selected, information for all components in that feature is available. If a package is selected, information for all features - and thus all components - in that package is available.

Detailed instructions for using the tabs for each category of Windows Installer Package data are presented in the chapters listed below.

Compression

General Package Options

Additional MSI Package Options

Variables and Properties

Package Conditions

Files

Shortcuts

Registry

System Services

File Edits

Advertising

Merge Modules

HOW TO PERFORM COMMON PACKAGE OPERATIONS

BUILD A PACKAGE

You can build Windows Installer packages, merge modules, transforms, and patches. You can build them manually or let the *Discover Wizard* step you through the process. You build packages manually in the Console. You use the *Discover Wizard* (*Discover.exe*) on a clean reference machine to build packages, merge modules, and transforms. You use the *Patch Wizard* on any machine to build a transform.

See Building Packages for detailed instructions on how to build Windows Installer packages using the *Discover Wizard*, how to build Windows Installer Patches using the *Patch Wizard*, and how to build all types of packages manually in the Console.

ADD PACKAGES

You can add any existing Windows Installer package to a list in the console, regardless of whether it was created with WinINSTALL, supplied by a software vendor, or created with other third-party tools. You cannot add a package directly beneath *Software Distribution Lists,* the top-level node, but you can add a package to any list nested beneath the top-level node.

To add an existing package to the Console, follow these steps.

HOW TO ADD A NEW PACKAGE TO THE CONSOLE

1. **In the tree pane, expand the *Software Distribution* node and the *Software Distribution Lists* sub-node.**
2. **Highlight the list or sub-list to which you want to add a new package and do one of the following:**

- **Right-click the list and choose *New->Windows Installer Package* from the pop-up menu.**
- **From the *File* menu on the menu bar, choose *New-> Windows Installer Package*.**

3 **On the *Create Package* dialog, do the following:**

- **For *Filename*, enter or browse to the name of the new package.**
- **For *Description*, enter a brief label for the package. This will display in the console.**
- **Click *OK*.**

You can now view and edit this package in the data view.

HOW TO ADD AN EXISTING PACKAGE TO THE CONSOLE

1 **In the tree pane, expand the *Software Distribution* node and the *Software Distribution Lists* sub-node.**

2 **Highlight the list or sub-list to which you want to add a package and do one of the following:**

- **Click the icon on the toolbar to add the desired package type:**

   **(for Windows Installer package)**

- **Right-click the list and choose *Add Existing->Windows Installer Package* from the pop-up menu.**
- **From the *Actions* menu on the menu bar, choose *Add Existing-> Windows Installer Package*.**

3 **On the *Open* dialog, browse to the package (.MSI file) to be added and double-click it.**

The dialog will close, and the selected package will display beneath the selected list file in the tree view of the console. You can now view and edit this package in the data view.

EDIT A PACKAGE

From the console, you can edit Windows Installer packages, whether they were created by WinINSTALL, an Independent Software Vendor (ISV), or any other third-party tool.

To edit a package in the Console, follow these steps:

1 **In the tree pane, expand the *Software Distribution* node and the *Software Distribution Lists* sub-node and select the package that you want to edit.**

2 **In the list pane, select the category of package information that you want to edit:**

- **General**
- **Files**
- **Shortcuts**
- **Registry**

- **Services**
- **Edits**
- **Advertising**

3 **In the data pane, select a tab or sub-tab and add or modify information.**

4 **Continue selecting categories in the list pane and tabs in the data pane until you have entered all of the necessary information.**

5 **Save the changes in the package by either selecting *Save* from the *File* menu or clicking the *Save* (diskette) icon on the tool bar.**

SEARCH AND REPLACE STRINGS IN A PACKAGE

WinINSTALL provides a simple facility to search for a character string within a package and replace it with another character string. For example, perhaps the package specifies that the installers are to copy all files to a directory called *OldDir*, but the files now need to go into a directory called *NewDir* instead. The following steps explain how to perform this operation.

1 **Select the desired package in the tree pane and select *Replace* from the *Edit* menu at the top of the console. The *Search and Replace* dialog will appear.**

2 **Under *Sections to Search*, check the checkboxes for sections of the package that you want to search. By default, all of the checkboxes are checked. (In the example given above, you would check only the *File Destination* checkbox).**

3 **In the *Replace All Occurrences …* textbox, type the string that you want WinINSTALL to find. (In the above example, you would enter *OldDir*).**

4 **In the *With This* textbox, type the new string that will be used to replace the string found above. (In the example given above, you would enter *NewDir*).**

5 **Click *OK*. WinINSTALL immediately searches the specified sections of the package and makes the appropriate replacements.**

***NOTE:** This technique does not replace character strings in a text file on the user's workstation. It only replaces strings in the package that will be processed on the user's workstation.*

5

BUILDING PACKAGES

WinINSTALL provides the means of building software distribution packages in all supported formats. This capability enables building all types of Windows Installer packages: msi files, Merge Modules, Transforms, and Patches. Windows Installer packages, Merge Modules and Transforms can all be built automatically, through the *Discover Wizard*, or manually, through the WinINSTALL Console. Building Windows Installer Patches is done by means of the *WinINSTALL Patch Wizard*. Building Windows Installer Patches is done by means of the *WinINSTALL Patch Wizard*.

This chapter details the steps involved in creating each of these package types, both automatically and manually.

USING DISCOVER TO BUILD PACKAGES AUTOMATICALLY

The *Discover Wizard* is a very flexible utility that uses snapshot technology to enable the automatic building of Windows Installer packages, merge modules, and transforms.

Building a package with *Discover* involves three basic steps:

1. **Run *Discover* to perform the *Before Snapshot*, which captures the state of a machine before the application has been installed. If you elect to use an *Archived Before Snapshot*, this step need be performed only once; Archived Before Snapshots can be reused repeatedly to build whatever type of package is desired.**
2. **Install the application.**
3. **Run *Discover* again to perform the *After Snapshot*, which captures the state of the machine after the application has been installed. *Discover* then compares the *Before* and *After* snapshots and builds the package based on the differences.**

The process of building a package using a *Basic Before Snapshot* differs from doing so using an *Archived Before Snapshot*, but only in the order of the screens. The same information is required in either case. The two variations on the process are covered separately, below.

LAUNCHING DISCOVER

You can either launch *Discover* from within the Console or from a clean machine (Discover Reference Machine) where no WinINSTALL components have been installed. In production, *Discover* is always run from a clean reference machine.

You can run *Discover* from the Console, but this capability exists solely to provide administrators with a convenient way to become familiar with the *Discover Wizard*. You would never actually build a production package on the console machine because, by definition, it is not clean if the WinINSTALL Console is installed.

FROM A CLEAN REFERENCE MACHINE

To run *Discover* from a reference machine, navigate to the *\bin* directory of the WinINSTALL share and execute *Disco.exe*.

FROM WITHIN THE WININSTALL CONSOLE

To run *Discover* from the WinINSTALL Console, select any node in the *Software Distribution* branch in the Console tree pane. Next, either click the *Discover* (magnifying glass) icon on the toolbar or select *Run->Discover ...* from the *File* menu on the menu bar. On the *Discover* dialog, navigate to *Disco.exe* in the *\bin* directory of the WinINSTALL share and click *OK*.

***WARNING:** It is not recommended that you run* Discover *to build production packages from within the WinINSTALL Console. The ability to launch* Discover *from the Console exists solely to provide administrators with a convenient way to become familiar with the* Discover Wizard*. You would never actually build a production package on the console machine because, by definition, it is not clean if the WinINSTALL Console is installed.*

DISCOVER REFERENCE MACHINE

A reference machine is a special machine that is used solely for the purpose of building packages. No WinINSTALL components are installed on a reference machine.

Technically, a reference machine should be a clean machine, with only the operating system and any necessary service packs installed. Because a reference machine does not need to be large and expensive, some administrators have two or more. In this way, they can create a package on one machine while returning a previously-used machine to its clean state. This practice reduces downtime - and the temptation to cut corners by using a dirty machine.

Alternatively, virtualization software products can simplify the process of returning a reference machine to its clean state. For instance, virtualization products allow you to create a virtual machine and save it in a clean state, build a package on it, and then discard all changes and return to the original clean state with a simple reboot. These virtual machines save a great deal of time because you can be certain of having a clean machine to use every

time, but you need install the operating system only once for each operating system used in your network.

ARCHIVED BEFORE SNAPSHOTS

To make the process even easier, *Discover* provides a shortcut in the form of an *Archived Before Snapshot*. This feature allows you to save time by taking a single *Before Snapshot* of the target machine and reusing it repeatedly to build multiple application packages of whatever type is desired.

***TIP:** Using an* Archived Before Snapshot *reduces the Discover process from three steps to two.*

A single Archived Before Snapshot *can be used to create multiple packages, including all supported package types.*

To create an *Archived Before Snapshot*, step through the following wizard panels:

WELCOME TO THE WININSTALL DISCOVER WIZARD

This welcome panel explains how the *Discover* process works.

Click *Next* to continue.

SELECTING SESSION

The *Selecting Session* panel enables selection of the type of *Discover* process to perform:

- *After* snapshot
- *Archived Before* snapshot
- *Before* snapshot

If you require full control over the *Discover* process, you can check the *Advanced Options* checkbox on this panel. Doing so will cause an additional four panels to appear during the sequence of Wizard Panels: *Advanced Options*, *File Exclusion Selection*, *Registry Exclusion Selection*, and *Text File Selection*. These panels and the options they present, are discussed in detail in the Discover Advanced Options section, below.

Select *Archived Before* and click *Next* to continue.

SELECTING BEFORE

On the *Selecting Before* panel, enter or browse to a path and filename for the *Archived Before* snapshot file you are about to create.

When you have specified the desired path and filename, click *Next* to continue.

SPECIFYING WORK DRIVE

On the *Specifying Work Drive* panel, select the drive where *Discover* will store its temporary work files. These files will be stored in a directory off the root and will be removed when *Discover* is finished.

TIP: Discover *work files can total several megabytes.*

Click *Next* to continue.

DRIVE SELECTION

On the *Drive Selection* panel, specify the drives to scan for changes when building the packages.

NOTE: *You can select as many drives as you like, but each drive requires time to scan. You should select only those drives where you expect that changes may occur.*

When you have specified the desired drive(s), click *Next* to continue.

ADVANCED OPTIONS

If you selected the *Advanced Options* checkbox on the *Selecting Session* panel at the start, the following advanced panels will appear at this point in the sequence:

- Advanced Options
- File Exclusion Selection
- Registry Exclusion Selection
- Text File Selection

These panels, and the options they present, are discussed in detail in the Discover Advanced Options section, below.

COMPLETING THE DISCOVER WIZARD

This dialog informs you that *Discover* is ready to create the *Before Snapshot*.

Click *Finish* to create the snapshot.

When *Discover* has finished processing, you receive a message that the *Archived Before* snapshot is complete.

BASIC BEFORE SNAPSHOTS

If you do not choose to create an *Archived Before Snapshot*, you will create a *Basic Before Snapshot* for each package you create with *Discover*. The process for creating such a snapshot varies slightly, depending on the type of package you are creating. Therefore, the details of creating a *Basic Before Snapshot* are covered in the section on using *Discover* to create each specific package type, below.

DISCOVER ADVANCED OPTIONS

Whether you are creating an *Archived Before Snapshot* or a *Basic Before Snapshot*, if you check the *Advanced Options* checkbox on the *Selecting Session* dialog of the *Before Snapshot*, the *Discover Wizard* will present four panels that would otherwise not appear:

- Advanced Options
- Exclusion File Selection
- Exclusion Registry Selection
- Selecting Text Files

The options presented on each of these panels is explained below.

ADVANCED OPTIONS

On the *Advanced Options* panel, you can focus the *Discover* process by selecting specific system areas to be scanned, and excluding other areas, instead of the default behavior of including the entire system. The following system areas can be included or excluded by checking or unchecking their respective checkboxes:

- File System
- Registry
- Icons/Shortcuts
- .INI Files
- Text Files
- File Attributes

- Hardware-specific registry areas.

WARNING: *Including* hardware-specific registry areas *is recommended only for very specific and unusual situations.*

If you do include hardware-specific registry areas in a Discover *process, you will need to edit the resulting package before distribution, to avoid making unintended changes to target machines.*

Failure to remove unsuitable hardware-specific entries could render a target system unusable.

EXCLUSION FILE AND REGISTRY SELECTION

In order to focus the *Discover* process on the relevant areas of an install, and to prevent unrelated, and potentially disruptive, system changes from being captured as part of the package, *Discover* uses file and registry exclusion lists. To make the *Discover* process as flexible as possible, these exclusion lists are editable.

The default exclusion lists are stored in the *\bin* directory of the WinINSTALL share, in the file *Discover.xml*.

The first time Discover is executed on a system, these exclusions are copied to the machine default lists, *Files.xcp* and *Registry.xcp*, in the local *Windows* directory. It is these files which are used to build the default exclusion list whenever *Discover* is run on the local system.

Each of the two *Exclusion Selection* panels presents the machine's default list of exclusions (as loaded from the local files *Files.xcp* and *Registry.xcp*).

Discover provides the ability to modify these exclusions, either on a per-session or a permanent basis, depending on the state of the *Machine's exclusions* checkbox.

If this checkbox is checked, any changes are saved to the machine's default exclusion list and will reappear as part of the default list the next time *Discover* is run on the same machine.

If the checkbox is left unchecked, any changes to the exclusion list will not be saved and will be used for the current *Discover* operation only.

***NOTE:** None of the changes you make in Discover affect the default exclusion list that is stored in the* Discover.xml *file on the WinINSTALL share. To make changes that will affect all machines which run Discover, you would have to manually edit* Discover.xml *in the* \bin *directory of the WinINSTALL share.*

To add an exclusion, click the *Add* icon to enter or browse to (*File Exclusions* only) the desired exclusion to add. Press *Enter* to add the new exclusion.

To modify an exclusion, double-click the entry in the list and make the desired changes, then press *Enter*.

For directory and registry key entries, make sure the entry includes a trailing backslash.

To delete an entry, select the desired entry and click the *Delete* icon.

To reset the list to the machine's default exclusion list, click the *Reset Contents* button.

TEXT FILE SELECTION

On the *Text File Selection* panel, you can specify text files whose contents you want to have scanned for differences.

The panel presents a default list of text files that would commonly be scanned, but you can add new files to this list or delete existing ones.

To add a new text file, click the *Ellipsis* button to browse to the desired file or enter the desired path and filename in the edit box and click the *Add to List* button.

To remove a file from the list, select the desired file in the list and click the *Remove from List* button.

USING DISCOVER TO BUILD A WINDOWS INSTALLER PACKAGE OR MERGE MODULE

To use *Discover* to build a Windows Installer package or Windows Installer Merge Module, you can either use an *Archived Before Snapshot* or create a *Basic Before Snapshot* before installing the application. The only difference between creating a package and a merge module is a radio button selection on the Selecting Target panel.

BUILDING A WINDOWS INSTALLER PACKAGE OR MERGE MODULE WITH A BASIC BEFORE SNAPSHOT

If you choose to run a *Basic Discover Before Snapshot*, the wizard gathers information from you on the following dialogs:

WELCOME TO THE WININSTALL DISCOVER WIZARD

This welcome dialog explains how the *Discover* process works.

Click *Next* to continue.

SELECTING SESSION

This dialog enables selection of the type of *Discover* process to perform:

- *After* snapshot
- *Archived Before* snapshot
- *Before* snapshot

If you require full control over the *Discover* process, you can check the *Advanced Options* checkbox on this panel. Doing so will cause an additional six panels to appear during the sequence of Wizard Panels. Two of these, *Discovering Transforms Query* and *Advanced MSI*, appear following the next panel and are discussed below. The other four, *Advanced Options*, *File Exclusion Selection*, *Registry Exclusion Selection*, and *Text File Selection*, appear later in the sequence and are discussed separately in the Discover Advanced Options section, above.

Select *Before snapshot* and click *Next* to continue.

DISCOVERING TRANSFORMS QUERY [ADVANCED OPTION]

This panel appears only if you checked the Advanced Options checkbox on the *Selecting Session* panel.

To create a Windows Installer package or Merge Module (not a Transform), be sure that the *Create Transform* checkbox is *unchecked*. The following items on this panel apply only to Transforms and will therefore remain disabled:

MSI File field
- Full path and file name of the MSI file to which the transform is to be applied.

Command line parameters field
- Command line parameters for the Windows Installer to use when installing the specified MSI file.

Make generated per-user components follow per-user or per-machine installs checkbox
- If per-user components are found, this checkbox indicates whether to enter them into the package so that they follow the AllUsers property and are installed as per-user if the package is installed per-user but are installed per-machine if the package is installed per-machine.

To create a Windows Installer Transform, please see Using Discover to Build a Windows Installer Transform, below.

Click the *Next* button to proceed.

ADVANCED MSI [ADVANCED OPTION]

On the Advanced MSI panel, you can specify the following options:

Schema Version
- Select desired MSI schema for the package.

Suppress advertising of COM registry entries
- If you will be distributing the package via Active Directory, it may be useful to suppress advertising COM registry entries.

Suppress auto-discovering merge modules
- If you have merge modules available in the Merge Module Cache, Discover will by default check the new package for the presence of any of these merge modules. You can disable that check by clicking this checkbox.

SELECTING TARGET

On the *Selecting Target* panel, enter the descriptive name of the package, which will identify the package in the console, and the path and file name of the package to be created.

Two radio buttons enable you to choose between creating a Windows Installer package or a Merge Module. This selection determines which type of package *Discover* will create in the end.

Select the desired language for the package being created from the drop-down on this panel.

Click *Next*.

LIST FILE SELECTION

On the *List File Selection* panel, you can specify a list file to which *Discover* will add the package being created. Enter the path and file name or browse to the desired list file.

Click *Next* to continue.

SPECIFYING WORK DRIVE

On the *Specifying Work Drive* panel, select the drive where *Discover* will store its temporary work files. These files will be stored in a directory off the root and will be removed when *Discover* is finished.

TIP: Discover *work files can total several megabytes.*

Click *Next* to continue.

DRIVE SELECTION

On the *Drive Selection* panel, specify the drives to scan for changes when building the packages.

NOTE: *You can select as many drives as you like, but each drive requires time to scan. You should select only those drives where you expect that changes may occur.*

When you have specified the desired drive(s), click *Next* to continue.

ADVANCED OPTIONS

If you selected the *Advanced Options* checkbox on the *Selecting Session* panel at the start, the following advanced panels will appear at this point in the sequence:

- Advanced Options
- File Exclusion Selection
- Registry Exclusion Selection
- Text File Selection

These panels, and the options they present, are discussed in detail in the Discover Advanced Options section, above.

COMPLETING THE DISCOVER WIZARD

This dialog informs you that *Discover* is ready to create the *Before Snapshot*.

Click *Finish* to create the snapshot.

When *Discover* has finished processing, you will receive a message that the *Before Snapshot* is complete. The message box will offer to run the application setup program. Click *OK* to be prompted for the setup program of the application that the package will install, or *Cancel* to run the setup program later.

WELCOME TO THE WININSTALL DISCOVER WIZARD

Next, install the application and make whatever changes you like to its settings.

***TIP:** Any changes you make to the application's settings, after installation but before running the* After Snapshot, *will automatically be captured as part of the installation package.*

After installing the application, run *Discover* again to perform the *After* snapshot and build the package.

At this point, the *Welcome* panel offers the choice of building the package (*Perform the After Snapshot*) or starting the process over (*Abandon the Before Snapshot*).

Select *Perform the After Snapshot* and click *Next* to build the package.

BUILDING A WINDOWS INSTALLER PACKAGE OR MERGE MODULE WITH AN ARCHIVED BEFORE SNAPSHOT

If you choose to build a Windows Installer package or Merge Module using an *Archived Before Snapshot*, when you perform the *After Snapshot* to build the package or Merge Module, the *Discover Wizard* gathers information from you on the following dialogs:

WELCOME TO THE WININSTALL DISCOVER WIZARD

This welcome dialog explains how the *Discover* process works.

Click *Next* to continue.

SELECTING SESSION

This dialog enables selection of the type of *Discover* process to perform:

- *After* snapshot
- *Archived Before* snapshot
- *Before* snapshot

If you require full control over the *Discover* process, you can check the *Advanced Options* checkbox on this panel. Doing so will cause two additional panels, *Discovering Transforms Query* and *Advanced MSI*, to appear following the next panel. Note that if you also checked the Advanced Options checkbox when creating the Archived Before Snapshot, you were presented with four additional advanced option panels during the process of creating the Before Snapshot: *Advanced Options*, *File Exclusion Selection*, *Registry Exclusion Selection*, and *Text File Selection*. These options are discussed separately in the Discover Advanced Options section, above.

Select *After snapshot* and click *Next* to continue.

DISCOVERING TRANSFORMS QUERY [ADVANCED OPTION]

This panel appears only if you checked the *Advanced Options* checkbox on the *Selecting Sessions* panel.

To create a Windows Installer package or Merge Module (not a Transform), be sure that the *Create Transform* checkbox is *unchecked*. The following items on this panel apply only to Transforms and will therefore remain disabled:

MSI File field
- Full path and file name of the MSI file to which the transform is to be applied.

Command line parameters field
- Command line parameters for the Windows Installer to use when installing the specified MSI file.

To create a Windows Installer Transform, please see Using Discover to Build a Windows Installer Transform, below.

Click the *Next* button to proceed.

ADVANCED MSI [ADVANCED OPTION]

On the Advanced MSI panel, you can specify the following options:

Schema Version
- Select desired MSI schema for the package.

Suppress advertising of COM registry entries
- If you will be distributing the package via Active Directory, it may be useful to suppress advertising COM registry entries.

Suppress auto-discovering merge modules
- If you have merge modules available in the Merge Module Cache, Discover will by default check the new package for the presence of any of these merge modules. You can disable that check by clicking this checkbox.

SELECTING TARGET

On the *Selecting Target* panel, enter the descriptive name of the package, which will identify the package in the console, and the path and file name of the package to be created.

Two radio buttons enable you to choose between creating a Windows Installer package or a Merge Module. This selection determines which type of package *Discover* will create in the end.

Select the desired language for the package being created from the drop-down on this panel.

When you have made the desired selections, click *Next* to proceed.

LIST FILE SELECTION

The *List File Selection* panel does not appear if you have chosen to create a Merge Module. This panel appears only if you have elected to create a Windows Installer package.

Here you have the option of selecting a list file to which *Discover* will add the package being created. Enter the path and file name or browse to the desired list file.

Click *Next* to continue.

COMPLETING THE DISCOVER WIZARD

This dialog informs you that *Discover* is ready to create the package or Merge Module.

Click *Finish* to create the After snapshot and produce the package or Merge Module.

When *Discover* has finished processing, you will receive a message that the package or Merge Module has been created.

USING DISCOVER TO BUILD A WINDOWS INSTALLER TRANSFORM

To use *Discover* to build a Windows Installer Transform, you can either use an *Archived Before Snapshot* or create a *Basic Before Snapshot* before installing the application.

The steps *Discover* uses to build a Transform are similar to those for building a package or Merge Module, but the process is a little more complicated. To build a Transform, *Discover* first requires the original MSI file to be installed on the *Discover* reference machine. During the *Discover* process, you will be asked to indicate whether or not the MSI file has been installed; if it has not been installed, *Discover* will install it then.

With the original MSI file installed, *Discover* takes a Before Snapshot of the system. Once the Before Snapshot is complete, you make whatever changes to the installation you want to be included in the Transform and then run *Discover* again to create the After Snapshot.

Internally, *Discover* will make a copy of the original MSI file, create a Merge Module out of the changes, apply the Merge Module to the copy of the MSI file, and then compare the original MSI file to the copy with the Merge Module applied. The differences between these two MSI files will be used to create the Transform.

BUILDING A WINDOWS INSTALLER TRANSFORM WITH A BASIC BEFORE SNAPSHOT

If you choose to run a *Basic Discover Before Snapshot*, the wizard gathers information from you on the following dialogs:

WELCOME TO THE WININSTALL DISCOVER WIZARD

This welcome dialog explains how the *Discover* process works.

Click *Next* to continue.

SELECTING SESSION

This dialog enables selection of the type of *Discover* process to perform:

- *After* snapshot
- *Archived Before* snapshot
- *Before* snapshot

To create a Windows Installer Transform, you must check the *Advanced Options* checkbox on this panel. Doing so will cause an additional panels to appear during the sequence of Wizard Panels, including the *Discovering Transforms Query*, which appears following the next panel and which allows you to specify that the Discover output should be a Transform. Four additional advanced options panels, *Advanced Options*, *File Exclusion Selection*, *Registry Exclusion Selection*, and *Text File Selection*, appear later in the sequence and are discussed separately in the Discover Advanced Options section, above.

Select *Before snapshot* and click *Next* to continue.

DISCOVERING TRANSFORMS QUERY [ADVANCED OPTION]

The *Discovering Transforms Query* panel will not appear unless you have checked the *Advanced Options* checkbox on the *Selecting Session* panel. The *Discovering Transforms Query* panel is where you instruct *Discover* to produce a Transform as the output file.

To create a Windows Installer Transform, select the *Create Transform* checkbox. Doing so will enable the following items on this panel:

MSI File field
- Full path and file name of the MSI file to which the transform is to be applied.

Command line parameters field
- Command line parameters for the Windows Installer to use when installing the specified MSI file.

If the MSI file is not already installed, *Discover* will install the MSI file as soon as you click the *Next* button.

INSTALLING MSI FILE

This panel appears during the installation of the MSI file (if *Discover* finds that it has not already been installed when you click *Next* on the *Discovering Transforms Query* panel).

When the installation is complete, click the *Next* button.

If the installation requires a reboot, after the reboot, run *Discover* again and follow the same steps.

SELECTING TRANSFORM ERROR

The *Selecting Transform Error* panel allows selection of error suppression flags to aid in applying the Transform.

Note that a Transform is built by comparing the installed state of two MSI files.The Transform is a database of changes to apply to the original MSI file to make its installation match the installation of the other.

When applying such changes, a predictable array of potential errors can result. When you apply a Transform through WinINSTALL, you are provided with the option to suppress certain of these errors. You can also build into the Transform itself the instruction to ignore selected errors and not report them.

***NOTE:** It is not recommended practice to suppress errors unless you know ahead of time that they will occur and are not concerned about the particular errors you have decided to suppress.*

The following error suppression checkboxes are available on the *Selecting Transform Error* panel:

- Ignore the addition of existing rows.
- Ignore the deletion of missing rows.
- Ignore the changing of missing rows.
- Ignore the addition of existing tables.
- Ignore the deletion of missing tables.
- Ignore difference between code pages.

Make any desired selections and click *Next* to continue.

SELECTING TRANSFORM VALIDATION

The *Selecting Transform Validation* panel allows selection of validation flags to aid in applying the transform.

When a Transform is applied to a Windows Installer package, version comparisons can be executed, and if the versions do not match, the application of the Transform fails. However, under some circumstances, you may want the versions of a Transform and MSI file to be considered valid even if they do not exactly match.

On the *Selecting Transform Validation* panel, you can specify exactly what matches (if any) will be required for the Transform to be applied successfully.

The following validation options are available:

The version radio buttons enable you to select which versions to compare:

- Major version only.
- Major and Minor versions.
- Major, Minor, and Update versions.
- Don't check (default).

If anything other than *Don't Check* is selected, the following radio buttons are also enabled, to specify the required relationship between the Transform version and the original package version:

- < original package.
- <= original package.
- = original package (default).
- >= original package.
- > original package.

The following additional validation checkboxes are also available:

- Default language must match.
- Product code must match.
- Upgrade code must match.

TARGET TRANSFORM

On the *Target Transform* panel, enter the path and file name of the Transform (.mst file) to be created.

SPECIFYING WORK DRIVE

On the *Specifying Work Drive* panel, select the drive where *Discover* will store its temporary work files. These files will be stored in a directory off the root and will be removed when *Discover* is finished.

TIP: Discover *work files can total several megabytes.*

Click *Next* to continue.

DRIVE SELECTION

On the *Drive Selection* panel, specify the drives to scan for changes when building the packages.

***NOTE:** You can select as many drives as you like, but each drive requires time to scan. You should select only those drives where you expect that changes may occur.*

When you have specified the desired drive(s), click *Next* to continue.

ADVANCED OPTIONS

If you selected the *Advanced Options* checkbox on the *Selecting Session* panel at the start, the following advanced panels will appear at this point in the sequence:

- Advanced Options
- File Exclusion Selection
- Registry Exclusion Selection
- Text File Selection

These panels, and the options they present, are discussed in detail in the Discover Advanced Options section, above.

COMPLETING THE DISCOVER WIZARD

This dialog informs you that *Discover* is ready to create the *Before Snapshot*.

Click *Finish* to create the snapshot.

When *Discover* has finished processing, you will receive a message that the *Before Snapshot* is complete.

***NOTE:** The completion message box will offer to run the application setup program. Because you are creating a Transform, rather than a package, no setup program is needed. Just click the* Cancel *button to dismiss the dialog.*

WELCOME TO THE WININSTALL DISCOVER WIZARD

Next, make whatever changes you like to the installation. These changes will comprise the Transform.

***TIP:** Any changes you make to the application's settings, after installation but before running the* After Snapshot, *will automatically be captured as part of the Transform.*

After making the desired changes, run *Discover* again to perform the *After* snapshot and build the Transform.

At this point, the *Welcome* panel offers the choice of building the package (*Perform the After Snapshot*) or starting the process over (*Abandon the Before Snapshot*).

Select *Perform the After Snapshot* and click *Next* to build the Transform.

BUILDING A WINDOWS INSTALLER TRANSFORM WITH AN ARCHIVED BEFORE SNAPSHOT

If you choose to build a Windows Installer Transform using an *Archived Before Snapshot*, when you perform the *After Snapshot* to build the Transform, the *Discover Wizard* gathers information from you on the following dialogs:

WELCOME TO THE WININSTALL DISCOVER WIZARD

This welcome dialog explains how the *Discover* process works.

Click *Next* to continue.

SELECTING SESSION

This dialog enables selection of the type of *Discover* process to perform:

- *After* snapshot
- *Archived Before* snapshot
- *Before* snapshot

To create a Windows Installer Transform, you must check the *Advanced Options* checkbox on this panel.

Selecting *Advanced Options* will cause the *Discovering Transforms Query* panel to appear following the next panel. Selecting *Advanced Options* is required to create a Transform, because it is on the *Discovering Transforms Query* panel that you instruct *Discover* to produce a Transform as the output file.

Note that if you also checked the *Advanced Options* checkbox when creating the *Archived Before Snapshot*, you were presented with four additional advanced option panels during the process of creating the Before Snapshot: *Advanced Options*, *File Exclusion Selection*,

Registry Exclusion Selection, and *Text File Selection*. These options are discussed separately in the Discover Advanced Options section, above.

Select *After snapshot* and click *Next* to continue.

DISCOVERING TRANSFORMS QUERY [ADVANCED OPTION]

The *Discovering Transforms Query* panel will not appear unless you have checked the *Advanced Option*s checkbox on the *Selecting Session* panel. This selection is required to create a Transform, because it is on the *Discovering Transforms Query* panel that you instruct *Discover* to produce a Transform as the output file.

To create a Windows Installer Transform, select the *Create Transform* checkbox. Doing so will enable the following items on this panel:

MSI File field
- Full path and file name of the MSI file to which the transform is to be applied.

Command line parameters field
- Command line parameters for the Windows Installer to use when installing the specified MSI file.

If the MSI file is not already installed, *Discover* will install the MSI file as soon as you click the *Next* button.

INSTALLING MSI FILE

This panel appears during the installation of the MSI file (if *Discover* finds that it has not already been installed when you click *Next* on the *Discovering Transforms Query* panel).

When the installation is complete, click the *Next* button.

If the installation requires a reboot, after the reboot, run *Discover* again and follow the same steps.

SELECTING TRANSFORM ERROR

The *Selecting Transform Error* panel allows selection of error suppression flags to aid in applying the Transform.

Note that a Transform is built by comparing the installed state of two MSI files.The Transform is a database of changes to apply to the original MSI file to make its installation match the installation of the other.

When applying such changes, a predictable array of potential errors can result. When you apply a Transform through WinINSTALL, you are provided with the option to suppress certain of these errors. You can also build into the Transform itself the instruction to ignore selected errors and not report them.

***NOTE:** It is not recommended practice to suppress errors unless you know ahead of time that they will occur and are not concerned about the particular errors you have decided to suppress.*

The following error suppression checkboxes are available on the *Selecting Transform Error* panel:

- Ignore the addition of existing rows.
- Ignore the deletion of missing rows.
- Ignore the changing of missing rows.
- Ignore the addition of existing tables.
- Ignore the deletion of missing tables.
- Ignore difference between code pages.

Make any desired selections and click *Next* to continue.

SELECTING TRANSFORM VALIDATION

The *Selecting Transform Validation* panel allows selection of validation flags to aid in applying the transform.

When a Transform is applied to a Windows Installer package, version comparisons can be executed, and if the versions do not match, the application of the Transform fails. However, under some circumstances, you may want the versions of a Transform and MSI file to be considered valid even if they do not exactly match.

On the *Selecting Transform Validation* panel, you can specify exactly what matches (if any) will be required for the Transform to be applied successfully.

The following validation options are available:

The version radio buttons enable you to select which versions to compare:

- Major version only.
- Major and Minor versions.
- Major, Minor, and Update versions.
- Don’t check (default).

If anything other than *Don’t Check* is selected, the following radio buttons are also enabled, to specify the required relationship between the Transform version and the original package version:

- < original package.
- <= original package.
- = original package (default).
- >= original package.
- > original package.

The following additional validation checkboxes are also available:

- Default language must match.
- Product code must match.
- Upgrade code must match.

TARGET TRANSFORM

On the *Target Transform* panel, enter the path and file name of the Transform (.mst file) to be created.

COMPLETING THE DISCOVER WIZARD

This dialog informs you that *Discover* is ready to perform the *After Snapshot*.

Click *Finish* to perform the snapshot and create the Transform.

When *Discover* has finished processing, you will receive a message that the Transform has been created.

EMBEDDING TRANSFORMS

After you have created a transform, you may want to embed it into a Windows Installer package. Doing so makes sure that the transform is always available when the package is processed. If the transform is very large, however, or if you have a number of transforms for a single package, you might prefer not to embed them. In such a case, you can provide them to your users as individual .mst files.

To embed a transform into a package, follow these steps:

1. **Navigate to the package under *Software Distribution Lists* in the tree pane.**
2. **Right-click the package and select *Transforms* and then *Embed* from the context menu.**
3. **The *Embed Transform* dialog will display. Click *Add Transform*.**
4. **The *Add Transform* dialog will display, allowing you to select the transform to embed.**
5. **Select the desired transform file (.mst) and click *OK*.**
6. **On the *Embed Transform* dialog, click *Embed*.**

APPLYING A TRANSFORM

Transforms are applied during installation.

To use the Windows Installer to install a package with a transform that is not embedded, you would enter a command such as the following:

```
MSIEXEC.EXE /I \\<server>\<share>\vendor.msi
TRANSFORMS=custom.mst
```

In this example, *custom.mst* is the desired transform, representing the changes to be applied to *vendor.msi*.

BUILDING PACKAGES MANUALLY

Building packages manually in WinINSTALL is essentially a process of creating an empty package of the desired type and then using the editing capabilities of the WinINSTALL Console to add whatever contents are desired.

This section provides step by step instructions on using the WinINSTALL Console to manually create WinINSTALL packages, Windows Installer Packages, Merge Modules, and Transforms.

The details of manually editing packages in the WinINSTALL Console are covered in the following chapters:

Compression

General Package Options

Additional MSI Package Options

Variables and Properties

Package Conditions

Files

Shortcuts

Registry

System Services

File Edits

Advertising

Editing Merge Modules is covered separately, in the Merge Modules chapter.

BUILD A WINDOWS INSTALLER PACKAGE MANUALLY

To build a package manually:

1. **In the tree pane of the Console, expand the *Software Distribution* node and the *Software Distribution Lists* sub-node.**
2. **Select a list and do either of the following:**
   - **Right-click the list and choose *New -> Windows Installer Package* from the pop-up menu.**
   - **Choose *New -> Windows Installer Package* from the *File* menu.**
3. **On the *Create Package* dialog, enter a name and description for the package.**
4. **On the next dialog, accept or change the language and the MSI schema version.**
5. **With the package highlighted in the tree pane, select a category in the list pane and enter information on the tabs in the data pane. The tabs change, depending on which category you select in the list pane. See the various chapters, listed above, for details on manually editing a Windows Installer package.**
6. **When you have finished entering information, save your changes by clicking the *Save* (diskette) icon on the toolbar or choosing *File --> Save* from the menu bar.**

BUILD A WINDOWS INSTALLER MERGE MODULE MANUALLY

1. **In the tree pane of the Console, expand the *Software Distribution* node and the *Merge Module Folders* sub-node.**
2. **Select the desired Merge Module folder and do one of the following:**
   - **Right-click the folder and choose *Add Merge Module* from the pop-up menu.**
   - **Click the *Add Merge Module* icon on the tool bar.**
   - **Choose *Add Merge Module* from the *File* menu.**
3. **On the *Create Package* dialog, enter a file name and description for the Merge Module.**
4. **On the next dialog, accept or edit the module name, language and MSI schema version. When you click *OK*, the Merge Module will be added to the selected folder in the Merge Module Cache.**
5. **To add content to the Merge Module, the first step is to expand the Merge Module in the tree pane, so that the generic *All Components* sub-node appears.**
6. **Right-click the *All Components* sub-node and select *Add Component*. A new component will appear beneath the *All Components* sub-node.**
7. **With the Merge Module highlighted in the tree pane, select a category in the list pane and enter information on the tabs in the data pane. The tabs change, depending on which category you select in the list pane. See the Merge Modules chapter for details on manually editing a Windows Installer Merge Module.**

8 **When you have finished entering information, save your changes by clicking the *Save* (diskette) icon on the toolbar or choosing *File* --> *Save* from the menu bar.**

BUILD A WINDOWS INSTALLER TRANSFORM MANUALLY

Building a Transform in the Console is as simple as making edits to an existing Windows Installer package (*.msi file) and saving those changes as a Transform. Before making these edits, though, you need to let the Console know that you intend to save the changes as a Transform, rather than as an updated version of the existing MSI file.

The steps to build a transform manually are as follows:

1 **In the tree pane of the Console, expand the *Software Distribution* node and select the desired Windows Installer package.**

2 **Next, do one of the following:**

- **Right-click the package and choose *Transforms -> Start/Finish* from the pop-up menu.**
- **Choose *Transforms -> Start/Finish* from the *Actions* menu.**

3 **The *Start Creating Transform?* dialog appears, providing a summary of how creating a Transform works. click *OK* to begin making the desired changes.**

4 **Edit the selected Windows Installer package to reflect all the desired changes.**

5 **To save the changes to a Transform file, do any one of the following:**

- **Click the *Save* (diskette) icon on the tool bar.**
- **Select *Save* from the *File* menu.**
- **Right-click the package and choose *Transforms -> Start/Finish* from the pop-up menu.**
- **Choose *Transforms -> Start/Finish* from the *Actions* menu.**

6 **The *Save Transform* dialog appears. Enter the desired filename (*.mst) for the Transform to be created and click *OK*.**

***NOTE:** Microsoft advises users to not add resources to existing components when creating a Transform. Instead, the recommended practice is to create new components (and possibly features as well) and add the new resources to those.*

BUILDING WINDOWS INSTALLER PATCHES

A Windows Installer patch (.msp file) is a special Windows Installer package file which updates an existing installation to a later version (for example, moving from version 1 of an application to version 2 of the same application).

WinINSTALL can easily create a Windows Installer patch by comparing an updated application package (called the *upgraded image*) to the original package (called the *target image*) of the same application. WinINSTALL compares the two packages and produces a *patch package* (.msp file), which can be applied to machines which have previously installed the original package (*target image*). Once the patch package is installed, the application installed on the machines will match that which would have been installed by the updated package (upgraded image).

THE WININSTALL PATCH WIZARD

To enable creation of Windows Installer patches (.msp files), WinINSTALL provides the *Patch Wizard*, which makes creating a patch a very simple operation.

You simply invoke the wizard from the WinINSTALL Console, supply the information it requests and click the *Next* button on each page. When you click the *Finish* button on the final page, you have a software patch that is ready to use.

To launch the *WinINSTALL Patch Wizard*, select the *Software Distribution* node or any item beneath it in the Console tree pane then select *Patch Wizard* from the *File->Run* menu.

The *Welcome to the WinINSTALL Patch Wizard* panel will appear, explaining the purpose and operation of the patch creation process. When you click the *Next* button, you will begin the process of creating the patch, involving the following panels:

DEFINING THE PATCH PACKAGE

On the *Defining the Patch Package* panel, you must provide the following information:

.msp path and filename
- Full path and filename of the patch file to be created.

Log file path and filename
- Full path and filename for the (optional) text log file for the patch creation process.

GUID
- Click the *Ellipsis* icon to generate a unique GUID for the patch.

Click *Next* to continue.

SELECTING APPLICATION VERSIONS

On the *Selecting Application Versions* panel, you specify the two Windows Installer packages (.msi files) that constitute the old (*target image*) and new (*upgraded image*) versions of the application.

***TIP:** A single Microsoft Patch Package (.msp file) can include upgrades from multiple old versions (target images).*

To add an *upgraded image* (new version), right-click the Patch Package icon and select Add Upgraded MSI. Browse to and select the desired upgraded (new version) .msi file on the *Open* dialog.

To add a target image (old version), right-click the upgraded package that you added above and select *Add Target MSI*. Browse to and select the appropriate target (old version) .msi file on the ***Open*** dialog.

To add additional target images, simply repeat step 2, above.

When you have added all desired upgraded and target images, click *Next* to continue.

COMPLETING THE PATCH WIZARD

The *Completing the Patch Wizard* panel informs you that the wizard has gathered all of the information it needs and is ready to build the patch package for you. At this point, WinINSTALL has already built a patch creation package (.pcp file) and will now use that file to create the actual patch package.

The *Advanced* button provides access to a set of optional properties that can prevent certain problems from interfering with the creation of the package. This button also enables access to a table view of the data in the .pcp file through the WinINSTALL MSI Table Editor.

Click *Finish* to build the patch.

ADVANCED OPTIONS

If you click the *Advanced* button on the *Completing the Patch Wizard* panel, the *Advanced Options* panel appears, providing checkboxes to set the following four properties to override potential problems that could prevent the successful creation of the patch package:

- Allow product code to differ
- Allow product major version to differ
- Include whole file instead of creating a binary file patch

- Ignore missing files from target images

Click the *View Tables* button to view and modify the information in the .pcp file before creating the patch. The *MSI Table Editor* dialog provides a list of all of the tables in the .pcp database and allows you to view and modify the actions, descriptions, and templates in each.

Click *Finish* on the *Completing the Patch Wizard* panel to build the patch.

6

Package Validation and Repair

WinINSTALL can perform a contextual validation of a Windows Installer package or merge module from within the Console, and it can also automatically repair many detected problems, if desired. Internal consistency evaluators, or ICEs, scan the selected package for database entries that - while individually valid - may be problematic when viewed as part of the whole database. (Remember that a Windows Installer package is a database.) ICEs are custom actions written in VBScript, JScript or as a DLL or EXE. Some ICEs perform syntactical checks on the database (for example, if one table in the database contains a pointer to a row in another table, does that row exist?). Other ICEs perform semantic checks that look at database relationships in a broader view.

ICEs are stored in a special .msi database called a .cub file. WinINSTALL includes four .cub files that can be used to validate an MSI file - *darice.cub*, *logo.cub*, *xplogo.cub*, and *mergemod.cub*. The file *darice.cub* contains all the standard, Microsoft-supplied ICEs used to validate Windows Installer packages. *Logo.cub* and *xplogo.cub* are subsets of *darice.cub* intended for specialized use. The file *mergemod.cub* contains the Microsoft-supplied ICEs used to validate Windows Installer merge modules. When you validate a package in WinINSTALL, you can use these standard .cub files, or you can create your own.

ICE validation merges the .cub file and your database into a third, temporary database. The Windows Installer is then run against that temporary database, displaying warnings, errors, debugging information, and API errors as it executes each ICE in the .cub file. When the installer finishes executing the ICEs, it closes the .msi file, .cub file, and temporary database without saving any changes. The original .msi file and .cub file remain unchanged by the validation process.

When you perform a validation, four types of messages may be returned.

Errors

- Report database authoring that causes incorrect behavior. For example, duplicate component GUIDs cause the installer to incorrectly register components.

Warnings

- Report database authoring that causes unexpected side-effects or incorrect behavior in certain cases. For example, entering the same property name in two conditions that differ only by the case of letters in the name. Because the installer is case-sensitive, the installer treats these as different properties.

Failures • Report failure of an ICE custom action, often the result of such severe database authoring problems that the ICE cannot even run.

Informational • Messages provide information about an ICE, rather than indicating a database authoring problem. For example, they can provide information about the ICE itself, such as a brief description, or provide progress information as the ICE runs.

An ICE message commonly returns a range of information.

- Name of the ICE that has found an error
- Date the ICE was created
- Author of the ICE
- Date the ICE was last modified
- Description of the API error causing the ICE to fail
- Description of the error
- A warning to the user
- Name of the database table containing the error or warning
- Name of the table column containing the error or warning
- Primary keys of the table containing the error or warning
- A URL to an HTML file giving debugging suggestions
- A string that can contain other information

***TIP:** You only need to validate those Windows Installer packages that you create when you repackage a non-MSI application using Discover. Vendor-supplied MSI applications should not be validated as the vendor-supplied MSI should be guaranteed to work in the form distributed.*

VALIDATING A WINDOWS INSTALLER PACKAGE

HOW TO VALIDATE A WINDOWS INSTALLER PACKAGE

1 **In the tree pane, expand the *Software Distribution* node and the *Software Distribution Lists* sub-node.**

2 **Navigate to the desired Windows Installer package and do one of the following:**

- Right-click the package and choose *Validate Package* from the pop-up menu.
- Choose *Validate Package* from the *Actions* menu on the menu bar at the top of the screen.

3 **On the *MSI Package Validator* dialog, select the .cub file to use for validation, select specific ICEs you want included, and click GO.**

***NOTE:** Performing a validation does not alter the package or the merge module being validated or the .cub file containing the ICEs. To make changes needed to correct problems, see the next section.*

REPAIRING A VALIDATED WINDOWS INSTALLER PACKAGE

Once you have validated a Windows Installer package or merge module, you can select specific ICEs for WinINSTALL to repair automatically. If the error can be repaired automatically, WinINSTALL will do so; if not, you will be notified that repair is not possible.

HOW TO REPAIR A VALIDATED PACKAGE

1 **After validating a Windows Installer package, while the *MSI Package Validator* dialog is still open, select the ICE errors you want repaired.**

2 **On the *MSI Package Validator* dialog, click *Repair*.**

7

CONFLICT ASSESSMENT

WinINSTALL enables you to check for potential conflicts among individual packages, among individual packages and the contents of list files, or among packages and a specified "baseline" configuration. This capability, called *Conflict Assessment*, lets you diagnose potential problems before you actually perform an installation.

To check for potential conflicts, you first define a conflict assessment that includes one or more packages and, optionally, a baseline. A baseline represents the initial state of a machine before any packages are installed, for example, a machine with only a particular operating system and service pack installed. Once the conflict assessment has been defined, the next step is to run it. If your conflict assessment includes only packages or packages and a list file, the contents of all the packages are compared to predict whether installation of the packages together will cause conflicts. If your conflict assessment also includes a baseline, the contents of all the packages are compared to each other and to the baseline, to determine whether the packages cause a conflict when installed on the baseline system.

If you plan to use a baseline in a conflict assessment, it is recommended that you first run a conflict assessment that includes only the baseline. This action will populate the database with all of the information from this baseline, a time-consuming process. Once this initial population has been completed, using that baseline data in other conflict assessments is much faster.

You create a baseline on an actual machine with the desired configuration by executing a utility located in the *\bin* folder of the WinINSTALL share. You perform conflict assessments from the *Conflict Assessment* node of the tree pane of the WinINSTALL Console, where you can perform the following actions:

- Create a Conflict Assessment.
- Edit a Conflict Assessment.
- Delete a Conflict Assessment.
- Run a Conflict Assessment.
- View the results of a Conflict Assessment.

The first step is to create a baseline on a local machine. Next, the *Conflict Assessment Wizard* enables you to create a conflict assessment and to edit it, if desired. Once the conflict assessment has been created, you run the conflict assessment from the console. Finally, you view the results of the conflict assessment in the Console or in special WinINSTALL

reports, such as *Assessed Packages* and *Conflict Assessment Results*. Conflict assessment data is stored in the WinINSTALL database in a set of tables with the prefix *WICA_*.

HOW TO GENERATE A CONFLICT ASSESSMENT BASELINE

You can include baselines as well as packages in conflict assessments. A baseline defines the initial state of a machine before any packages are installed, such as a machine with only a particular operating system and service pack installed. You generate a baseline on such a machine, rather than on the Console machine.

1. **From the machine that you want to baseline, explore through the network to a WinINSTALL share.**
2. **Under the WinINSTALL share, expand the *bin* directory.**
3. **Navigate to *WIBaselineGen.exe* and double-click it.**
4. **On the WinINSTALL *Conflict Assessment Baseline Generator* dialog, specify whether to scan files, registry entries, or both. If you choose to scan files, indicate whether to include .INI file contents and shortcut definitions and specify the top-level folder where the file scan should begin.**
5. **A progress bar displays as the baseline is being generated and a message box tells you when generation is complete.**

THE CONFLICT ASSESSMENT WIZARD

You can launch the *Conflict Assessment Wizard* to create a conflict assessment in three ways:

- Select the *Conflict Assessment* node in the tree pane and select *New Assessment* from the *Conflict Assessments* menu.
- Select the *Conflict Assessment* node in the tree pane and click the *New* icon on the toolbar.
- Select the *Conflict Assessment* node in the tree pane and click the *New* button on the data pane.

The WinINSTALL *Conflict Assessment Wizard* presents the process of creating a conflict assessment in five steps.

WELCOME

The first panel prompts for a descriptive name for the conflict assessment you are creating.

BASELINE SELECTION

The *Baseline Selection* panel prompts you to enter or browse for an optional baseline file (*.bsl) to include in the conflict assessment.

See the section on How to Generate a Conflict Assessment Baseline, above for details on baselines.

PACKAGE SELECTION

The *Package Selection* panel presents a view of the software distribution and patch packages on the current WinINSTALL share. As in the Console tree pane, you can expand the base list files to view the packages and lists beneath them.

Each list and package has a checkbox. Click to check the boxes next to the desired packages and/or lists to include in the conflict assessment.

CONFIRMATION

The *Conflict Assessment Confirmation* panel presents a summary of the conflict assessment you are creating, including the name, the selected baseline file, and the selected packages and/or lists.

If any of the information is not as you intended, click the *Back* button to return to the earlier panels and make the needed corrections. If the information is correct, click the *Next* button to proceed.

COMPLETION

When you click *Finish* on the last dialog of the wizard, your conflict assessment is saved and displays in the *Conflict Assessments* tab on the data pane.

Because conflict assessments can take a long time to process, the *Do not run this Conflict Assessment right now* checkbox is checked by default, meaning the conflict assessment will not execute when you click the *Finish* button. But it will be saved for execution later from the *Conflict Assessments* tab on the data pane.

If you uncheck the *Do not run this Conflict Assessment right now* checkbox, the conflict assessment will execute immediately when you click the *Finish* button, and the results will be displayed in the *Conflict Assessment* window, which will pop up as soon as the conflict assessment is created.

HOW TO EDIT A CONFLICT ASSESSMENT

You can modify a conflict assessment in any of three ways:

- Select the *Conflict Assessment* node in the tree pane, highlight an existing conflict assessment in the data pane, and select *Edit Assessment* from the *Conflict Assessments* menu.
- Select the *Conflict Assessment* node in the tree pane, select an existing conflict assessment in the data pane, and click the *Edit* icon on the toolbar.
- Select the *Conflict Assessment* node in the tree pane, select an existing conflict assessment in the data pane, and click the *Edit* button on the data pane.

The Conflict Assessment Wizard opens. The existing conflict assessment information is used to pre-populate the wizard dialogs. Edit the information you want to change as you step through the pages of the wizard (described above). When you click *Finish* on the last dialog of the wizard, your modified conflict assessment is saved.

HOW TO DELETE A CONFLICT ASSESSMENT

You can delete an existing conflict assessment in any of three ways:

- Select the *Conflict Assessment* node in the tree pane, highlight the desired conflict assessment, and select *Delete Assessment* from the *Conflict Assessments* menu.
- Select the *Conflict Assessment* node in the tree pane, highlight the desired conflict assessment, and click the *Delete* icon on the toolbar.
- Select the *Conflict Assessment* node in the tree pane, highlight the desired conflict assessment, and click the *Delete* button on the data pane.

You will receive a message asking you whether you are sure that you want to delete the specified conflict assessment. Click *Yes* to continue the deletion process; click *No* to cancel it.

HOW TO RUN A CONFLICT ASSESSMENT

You can run a conflict assessment in any of three ways:

- Select the *Conflict Assessment* node in the tree pane, highlight the desired conflict assessment, and select *Run Assessment* from the *Conflict Assessments* menu.
- Select the *Conflict Assessment* node in the tree pane, highlight the desired conflict assessment, and click the *Run* icon on the toolbar.

- Select the *Conflict Assessment* node in the tree pane, highlight the desired conflict assessment, and click the *Run* button on the data pane.

WinINSTALL processes your conflict assessment. Once processing has finished, the results display in a dialog and can be reviewed later by simply clicking a button or menu choice.

***TIP:** Depending on the number and size of the packages involved and choices you made when defining the conflict assessment, running the assessment may take a long time. You may want to plan this process for a time when you do not need to use the Console for other tasks.*

VIEWING CONFLICT ASSESSMENT RESULTS

Once you have run a conflict assessment, you can view the results either in the Console or in specific reports.

HOW TO VIEW THE RESULTS OF A CONFLICT ASSESSMENT IN THE CONSOLE:

1. **Select the *Conflict Assessment* node in the tree pane.**
2. **Highlight the desired conflict assessment in the data pane and do one of the following:**
   - Select *View Results* from the *Conflict Assessments* menu.
   - Click the *Results* icon on the toolbar.
   - Click the *Results* button on the data pane.

HOW TO VIEW THE RESULTS OF A CONFLICT ASSESSMENT IN A REPORT:

1. **Expand the *Reports* node in the tree pane.**
2. **Select the *Conflict Assessment* report category beneath it.**
3. **On the data pane, choose one of the listed reports. For some of the reports, you will be asked to supply a parameter to filter the data that is returned.**

CONFLICT ASSESSMENT CATEGORIES

The following sections list the categories of conflict, subcategories of conflict, and conflict messages presented by the Conflict Assessment feature.

MAIN CONFLICT CATEGORIES

2	• File conflict
3	• Registry conflict
4	• Shortcut conflict
5	• INI file conflict

FILE CONFLICT SUBCATEGORIES AND MESSAGES

1	• Both packages add a file and their attributes are different
2	• Both packages remove the same file
3	• First package adds file and second removes it
4	• Second package adds file and first removes it
5	• Both packages add a file and some attributes are unknown
6	• Both packages add a file and the attributes match
7	• The file names are the same but their path is not necessarily the same
8	• Both add the same directory
9	• Both remove the same directory
10	• First package adds directory and second removes it
11	• Second package adds and first package removes directory

REGISTRY CONFLICT SUBCATEGORIES AND MESSAGES

1	• Both packages add the same registry key
2	• Both packages remove the same registry key
3	• First package adds a registry key and second removes it
4	• Second package adds a registry key and first removes it
5	• Packages set the same registry value to different values
6	• Both packages remove the same registry value
7	• First package adds and second package removes value
8	• Second package adds and first removes value
9	• Both packages set the same registry value to the same value

SHORTCUT CONFLICT SUBCATEGORIES AND MESSAGES

1	• Both packages write a shortcut in the same place, but the details of the shortcut differ
2	• Both packages remove the same shortcut
3	• First package adds a shortcut and second removes it
4	• Second package adds a shortcut and first removes it
5	• Both packages add a shortcut and both shortcuts have the same attributes

INI FILE CONFLICT SUBCATEGORIES AND MESSAGES

1	• Both packages add the same ini section
2	• Both packages remove the same ini section
3	• First package adds and second package removes the same ini section
4	• Second package adds and first package removes the same ini section
5	• Both packages add the same ini value, with different values
6	• Both packages remove the same ini value
7	• First package adds and second package removes the ini value
8	• Second package adds and first package removes the ini value
9	• Both package add the same ini value with the same value

8

COMPRESSION

WINSTALL provides the capability of combining all the components of a package into one or more compressed files. This approach can make it easy to move packages from one location to another and can make it simple to prevent inadvertent changes to files which are part of the installation.

MSI PACKAGE COMPRESSION

To compress a Windows Installer package, select it in the tree view and either right-click to display the context menu or click on the ***Actions*** menu. Select *Compress . . .* from the menu to display the *MSI Compression* dialog, where you can make the necessary choices to compress the package.

The source files for a Windows Installer package can be compressed into one or more cabinet files. These cabinet files can be included in the .msi file or they can be stored externally, as separate files. Compressed files are inserted into a standard Cabinet file of the type created by the Microsoft *Makecab.exe* cabinet file creation tool.

MSI FILE COMPRESSION SETTINGS

On the *MSI Compression* dialog, you can specify several settings, each of which is explained below.

Location of Cab Files
- Select whether the compressed file will be internal to the .msi file or a separate file.

Number of Cab Files
- Select whether to store the compressed source files in one cabinet file or in multiple cabinet files.

Maximum Cabinet Size (Bytes)
- If you store the files in multiple cabinet files, you can specify the maximum size, in bytes, of each cabinet file.

Reserved Space for Digital Signing (Bytes)
- If you store the cabinet files externally, you can specify an amount of space, in bytes, to reserve for digitally signing the package.

Once you have made your selections, click *Compress* to compress the expanded .msi file.

In the dialog's *Progress* indicator, WinINSTALL displays progress messages while a package is being compressed, ending with the message *"Finished."* At the end, the *Results*

field on the dialog will display the results of the compression action, letting you know whether or not the package was successfully compressed.

Click *Close* to continue working in the Console.

DIGITAL SIGNATURES

Windows Installer Packages with external .cab files may have digital signatures applied as a security feature. When the .msi file is installed, it will verify the integrity of the .cab files before installation, to assure that the .cab files have not been tampered with or corrupted in any way.

To take advantage of this WinINSTALL feature, you must have a digital certificate, you must have a Windows Installer package with one or more external cabinet (.cab) files, and you must use special utilities to sign the external .cab file(s). Once you have signed the .cab file(s), you can then use WinINSTALL to acquire, or register, the .cab file signatures into the .msi file, so that installing the .msi file will depend on a successful verification of the integrity of the .cab file(s).

APPLYING A DIGITAL SIGNATURE TO A WINDOWS INSTALLER PACKAGE

Applying a digital signature to a .msi file requires three separate steps.

1. **First, create an external .cab file, making sure to reserve an appropriate amount of space for the digital signature (see the MSI Package Compression section, above).**
2. **Next, use your digital certificate and special utilities to sign the .cab file (see Signing an External Cabinet File, below).**
3. **Finally, apply the .cab file's signature to the .msi file using the following procedure:**
   - **Select the package in the WinINSTALL Console.**
   - **Right-click and select *Signing Tables* from the context menu.**
   - **The *Digital Signing Tables* dialog will appear. Select the desired .cab file in the list. It should have a red *X* icon beside it, indicating that its signature has not been applied to the .msi file. Click the *Sign* button to acquire the signature from the .cab file and apply it to the .msi file. Click *OK* to dismiss the *Digital Signing Tables* dialog.**
   - **Save the package.**

The .msi file will now check the .cab file before installation to verify that it has not been tampered with or otherwise corrupted.

SIGNING AN EXTERNAL CABINET FILE

Signing a .cab file requires a certificate and special utilities to embed the necessary information into the signed package. The certification authority will have additional information on this procedure, as does Microsoft.

To begin, you will need a *.spc* file and a *.pvk* file. These files are obtained from a certification authority, such as Verisign. The *.spc* file contains your public key and other information, resides on your hard drive, and can be distributed to others.

The *.pvk* file contains a private key that corresponds to the public key in the *.spc* file. This file contains very sensitive information that should be closely guarded.

The following link contains helpful information concerning digital signing, including an example using the signing of a font file to illustrate the process:

http://www.microsoft.com/typography/developers/dsig/dsig.htm

The Microsoft Developer Network provides several useful utilities for digital signing. Information on these utilities can be found at the locations listed below.

signcode.exe:

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cptools/html/cpgrffilesigningtoolsigncodeexe.asp

makecert.exe:

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cptools/html/cpgrfcertificatecreationtoolmakecertexe.asp

cert2spc.exe:

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cptools/html/cpgrfsoftwarepublishercertificatetesttoolcert2spcexe.asp

9

VARIABLES AND PROPERTIES

Anyone who has ever filled out paperwork has used variables. When the form says *Name* and you enter *tjones* and I enter *bsmith*, we have assigned two different values to the same variable, *Name*. The major advantage of using variables in packages is that they allow you to customize packages for your particular environment or make them generic enough to run on almost any platform.

The Windows Installer natively supports a wide range of system variables – expressed as properties - which can be selected from an available list and inserted into a package. These system variables are not authored into the database (that is, they do not appear in the Property table for the MSI package if you view it) because they are *runtime properties*. *Runtime properties* are set only in memory (that is, they are set in the runtime version of the Property table) and are not persisted in the database.

HOW VARIABLES WORK

When you create a package with the *Discover Wizard*, the package that is created will contain certain references specific to the machine on which the package was created – even if you use a clean, quiet machine. Assume, for example, that you log on to a machine named *ReferencePC* with the username *Administrator* and build a package on that machine. Suppose that the Windows files on that machine reside in the directory *C:\WINNT*. The package you create will have references to a user called *Administrator*, to a machine called *ReferencePC*, and to a directory called *C:\WINNT*.

If you attempt to install this package on a machine with different values for the machine name, username, or Windows directory, you could have problems. To remedy this potential problem, you simply replace the PC-specific references in the package with variables. For example:

You can replace all occurrences of *C:\WIN* with the variable *[WindowsFolder]*.

Now, when the package is installed on a workstation, the installer interprets the variables at runtime and replaces them with the appropriate values for the local machine.

SYSTEM PROPERTIES

System properties let you tell the installer, "Whenever you see this property in a package, replace it with the value appropriate for the user running the package, as predefined by the Windows Installer."

SYSTEM PROPERTIES USED IN WINDOWS INSTALLER PACKAGES

The Windows Installer natively supports the following twenty-eight system properties (each followed by an example value from a typical system). At runtime, it automatically replaces these system properties with the appropriate value. You cannot modify these values.

[AdminToolsFolder]

- The full path to the file system directory that stores administrative tools.
- C:\Documents and Settings\All Users\Start Menu\Programs

[AdminToolsFolder]

- The full path to the file system directory that stores administrative tools.
- C:\Documents and Settings\All Users\Start Menu\Programs

[AppDataFolder]

- The full path to the file directory containing application data for the current user.
- C:\Documents and Settings\<user>\Application Data

[CommonAppDataFolder]

- The full path to the file directory containing application data for all users.
- C:\Documents and Settings\All Users\Application Data

[CommonFilesFolder]

- The full path to the Common Files folder for the current user.
- C:\Program Files\Common Files

[CommonFiles64Folder]

- The full path of the predefined 64-bit Common Files folder. (not used on 32-bit Windows)
- C:\Program Files\Common Files

[DesktopFolder]

- The full path to the current user's Desktop folder.
- C:\Documents and Settings\<user>\Desktop
- If an All Users profile exists and the [ALLUSERS] property is set, then this property is set to the full path to the Desktop folder for all users.
- C:\Documents and Settings\All Users\Desktop

[FavoritesFolder]

- The full path of the Favorites folder for the current user.

	• C:\Documents and Settings\<user>\Favorites
[FontsFolder]	• The full path of the system Fonts folder. • C:\WINNT\Fonts
[LocalAppDataFolder]	• The full path to the file system directory that serves as the data repository for local (non-roaming) applications. • C:\Documents and Settings\<user>\Local Settings\Application Data
[MyPicturesFolder]	• The full path to the MyPictures folder. • C:\Documents and Settings\<user>\My Documents\My Pictures
[PersonalFolder]	• The full path to the Personal folder for the current user. • C:\Documents and Settings\<user>\My Documents\Personal\
[PRIMARYFOLDER]	• The primary folder for the installation. • C:\ProgramFiles
[PrimaryVolumePath]	• The path of the volume designated by the [PRIMARYFOLDER] property. • For example, if the folder referenced by [PRIMARYFOLDER] resolves to C:\Program Files, then [PrimaryVolumePath] is set to C:.
[ProgramFilesFolder]	• The full path to the Program Files folder. • C:\Program Files
[ProgramFiles64Folder]	• The full path of the predefined 64-bit Program Files folder. • C:\Program Files
[ProgramMenuFolder]	• The full path to the Program Menu folder for the current user. • C:\Documents and Settings\<user>\Start Menu\Programs\
[ROOTDRIVE]	• The default drive for the destination directory of the installation. • The value for this property must end with '\'.
[SendToFolder]	• The full path to the SendTo folder for the current user. • C:\Documents and Settings\<user>\SendTo\
[SourceDir]	• The root directory containing the source cabinet file or the source file tree of the installation package.
[StartMenuFolder]	• The full path to the Start Menu folder. • C:\Documents and Settings\<user>\Start Menu\ • If an All Users profile exists and the [ALLUSERS] property is set, then this property is set to the full path to the Start Menu folder for all users.

	• C:\Documents and Settings\All Users\Start Menu\
[StartupFolder]	• The full path to the Startup folder. • C:\Documents and Settings\<user>\Start Menu\Programs\Startup\ • If an All Users profile exists and the [ALLUSERS] property is set, then this property is set to the full path to the Start Menu folder for all users. • C:\Documents and Settings\All Users\Start Menu\Programs\Startup\
[SystemFolder]	• The full path of the System folder. • C:\WINNT\System32.
[System16Folder]	• The full path to folder for 16-bit system DLLs. • C:\WINNT\System
[System64Folder]	• The full path to the predefined System64 folder. • C:\WINNT\System32
[TARGETDIR]	• The root destination directory for the installation. During an administrative installation, this property specifies the location to copy the installation package. During a non-administrative installation, this property specifies the root destination directory.
[TempFolder]	• The full path to the Temp folder. • C:\Temp
[TemplateFolder]	• The full path to the Template folder for the current user. • C:\Documents and Settings\<user>\Template\
[WindowsFolder]	• The full path to the Windows folder. • C:\WINNT
[WindowsVolume]	• The volume of the windows folder. This property always ends with a backslash.

ADDING SYSTEM PROPERTIES TO AN MSI PACKAGE

You manually insert system variables into an MSI package by browsing for or entering a system propertyin an eligible edit box. You can either place the cursor where you want to insert the variable in the edit box or replace existing text by highlighting a block of text in the edit box. You can click the browse button for the edit box and choose a system property from a list of available properties on a dialog.

USING THE SELECT DIRECTORY DIALOG TO ADD SYSTEM PROPERTIES TO AN MSI PACKAGE

When you click the ellipsis (browse) button for an eligible edit box, the various system properties display on the ***Select Directory*** dialog.

Property names prefixed with **%** represent system and user environment variables. These are never entered into the ***Property*** table. The permanent settings of environment variables can only be modified using the ***Environment*** table.

PROMPTING THE USER FOR A PROPERTY VALUE

WinINSTALL provides the ability to very quickly and easily generate a customized dialog to prompt the user during an install for a value to assign to a property. This property prompt dialog is based on a template MSI file, *QueryProperty.msi*, found in the *\bin* directory of the WinINSTALL share. See details on this feature in the Simple GUI section of the Additional MSI Package Options chapter of this Guide.

10

PACKAGE CONDITIONS

You can request that a package be processed only by clients that meet specific conditions. Launch Conditions and Feature Conditions, can be set for Windows Installer packages at the package and feature level.

If no conditions are set, the selected package can be installed on or removed from any computer by any user, unless other factors prevent the package from being installed or removed. But once a condition has been set, only those users or computers who meet all specified conditions will be able to install or remove the package.

WINDOWS INSTALLER LAUNCH CONDITIONS

Windows Installer packages and features can also impose special launch conditions, in addition to the types of conditions that can be set for NAI packages and lists. Such conditions must be met in order for the package or feature to be installed.

For each Windows Installer package launch condition, you must also specify an error message to display if the condition is not met.

Feature conditions do not include an error message. But they do include a numeric *Conditional Install Level* (see below for an explanation).

PACKAGE LAUNCH CONDITIONS

To set launch conditions on a Windows Installer package, select the package in the tree pane, select *General* in the list pane, select the *Advanced* tab in the data pane, and then select the *Launch Condition* sub-tab.

To add a condition to a package, click the *Add* icon and the *Launch Condition* dialog will display, providing a *Condition* field (accompanied by an ellipsis button), and an *Error Message* field. These fields are explained below.

To edit a condition, select the desired condition in the list and click the *Edit* icon to display the *Launch Condition* dialog.

To delete a condition, select the desired condition in the list and click the *Delete* icon.

LAUNCH CONDITION

This field contains the specification for the condition that must be met in order for the package to be installed. You can type the condition in manually or click the *Ellipsis* button

to display the *Condition* dialog (see description below), where you can construct the condition by selecting properties and operators from lists and drop-downs.

ERROR MESSAGE

This is the message that will display if the package launch condition is not met.

CONDITION DIALOG

On the *Condition* dialog (accessed by clicking the *Ellipsis* button on the *Launch Condition* dialog), select a property type from the drop-down on the left, then select the desired property from the list below the drop-down.

Next, select a property type from the drop-down on the right, then select the desired property from the list below the drop-down.

To complete the condition string, select an operator for the comparison operation from the drop-down between the two property lists.

Finally, select a logical operator to be used to combine this condition string with others and then click the *Add* button.

When you have added all the desired conditions, click *OK* to add the condition to the list on the *Launch Condition* tab.

***NOTE:** For a package to be installed, all specified* Launch Condition *logical comparisons must evaluate as* true.

FEATURE CONDITIONS

To set a condition for a Windows Installer feature, select the feature in the tree pane, select *General* in the list pane, and then select the *Conditions* tab.

To add a condition to a feature, click the *Add* icon and the *Feature Condition* dialog will display, providing a *Condition* field (accompanied by an ellipsis button), and an *Conditional Install Level* field. These fields are explained below.

To edit a condition, select the desired condition in the list and click the *Edit* icon to display the *Feature Condition* dialog.

To delete a condition, select the desired condition in the list and click the *Delete* icon.

CONDITION

This field contains the specification for the condition that must be met in order for the feature to be installed. You can type the condition in manually or click the *Ellipsis* button to display the *Condition* dialog (see description below), where you can construct the condition by selecting properties and operators from lists and drop-downs.

CONDITIONAL INSTALL LEVEL

This field contains a numeric value from 1 to 32,767. A feature is installed only if the feature level value is less than or equal to the current install level value. A conditional install value of 0 disables the feature from being installed. Typically, the install level is modified by options presented to the user during the install.

CONDITION DIALOG

On the *Condition* dialog (accessed by clicking the *Ellipsis* button on the *Feature Condition* dialog), select a property type from the drop-down on the left, then select the desired property from the list below the drop-down.

Next, select a property type from the drop-down on the right, then select the desired property from the list below the drop-down.

To complete the condition string, select an operator for the comparison operation from the drop-down between the two property lists.

Finally, select a logical operator to be used to combine this condition string with others and then click the *Add* button.

When you have added all the desired conditions, click *OK* to add the condition to the list on the *Conditions* tab.

***NOTE:** For a feature to be installed, all specified* Feature Condition *logical comparisons must evaluate as* true.

Section 3

Package Editing

Chapter 11: General Package Options

Chapter 12: Additional MSI Package Options

Chapter 13: Files

Chapter 14: Shortcuts

Chapter 15: Registry

Chapter 16: System Services

Chapter 17: Advertising

Chapter 18: File Edits

Chapter 19: Merge Modules

11

General Package Options

When you highlight a list or package in the tree pane and select *General* in the list pane, tabs display in the data pane.

This chapter discusses the *Summary* tab for packages, and all of the *General* tabs for MSI features and components.

When you highlight a package in the tree pane and select *General* in the list pane, a *Summary* tab displays in the data pane.

Summary Tab (Windows Installer Package)

On the *Summary* tab for an MSI package, you can view, enter, edit or delete administrative information about the highlighted package, such as product information, support information, and whether and how the application should display in *Add/Remove Programs* in the Control Panel.

The following items are available on this tab:

ARP Icon

Choose whether the application will appear on the *Add/Remove Programs* dialog (available from the Control Panel) and what actions will be available to the user if it does appear.

Show Product in ARP

When this checkbox is *not* checked, the application will not appear in the list of applications on the *Add/Remove Programs* dialog.

Show Change Button

When this checkbox is *not* checked, the user cannot modify the installation of the application from the *Add/Remove Programs* dialog.

Show Remove Button

When this checkbox is *not* checked, the user cannot uninstall the application from the *Add/Remove Programs* dialog.

***NOTE:** The ARP icon is associated with advertising an application.*

SUPPORT INFORMATION

The information entered in this field will display on the *Support Info* dialog when a user selects the application on the *Add/Remove Programs* dialog (available from the Control Panel) and then clicks *Click here for support information:*

CONTACT

This is the name of a person or entity that can be contacted for product support.

PHONE

This is the telephone number for product support.

ONLINE SUPPORT

This is the URL for online product support.

HELP FILE URL

This is the URL for the online help system.

UPDATE INFO URL

This is the URL where users can find update information about the product.

REGISTERED USER/COMPANY

This field identifies the user or company that is the registered owner of the product. This value can be obtained in one of three ways:

Automatic (get from Registry)
- The Windows Installer will automatically retrieve the name of the current user from the Registry.

Suppress
- The Windows Installer will not display this information during installation.

Use these values
- The Windows Installer will use the values entered on this dialog in the *User Name* and *Company Name* fields.

SHOW REPAIR BUTTON

When this checkbox is *not* checked, the *Repair* button does not appear on the *Add/Remove Programs* dialog.

MSI SCHEMA

When you are creating a new package, you should enter here the version of the MSI database schema that will be used in the creation of the package. For existing packages, the version is supplied automatically based on the MSI version installed on the PC where the application package was created, and it therefore cannot be changed on this tab.

VERSION

Enter the number that specifies the particular version of the product being installed. The version number is in the format -

`xxxxx.xxxxx.xxxxx.xxxxx`

where x represents a digit and the maximum version string allowed is *65535.65535.65535.65535*.

PRODUCT CODE

This field holds the *Globally Unique Identifier* (GUID) that is the principal identifier for the application or product. Letters in a product code GUID must be uppercase.

The 32-bit and 64-bit versions of an application's package must have different product codes.

If any 32-bit component of an application is recompiled into a 64-bit component, a new product code must be assigned.

***NOTE:** If significant changes are made to a product, then the product code should also be changed to reflect this. However, it is not a requirement that the product code be changed when the changes to the product are relatively minor.*

Click the *Ellipsis* button to browse for the product code. If no GUID exists, a message box will ask whether you want to generate a new GUID.

PACKAGE CODE

This field holds the GUID that is the principal identifier for the package.

***NOTE:** The package code must be unique for every package. A small update, a minor upgrade, or a major upgrade of the same application all require separate package codes. Versions of the same application in different languages also require separate package codes.*

Click the *Ellipsis* button to browse for the package code. If no GUID exists, a message box will ask whether you want to generate a new GUID.

PRODUCT LANGUAGE

From the drop-down list of available languages, you can select the language in which the application will display.

PRODUCT ID

This field contains the product identifier that will display when the application is selected in the *Add/Remove Programs* dialog.

MANUFACTURER

This field displays the name of the software vendor that created the application.

MANUFACTURER'S URL

This field holds the address of the web site for the software vendor that created the application.

COMMENTS

The contents of this field display when a user selects the application on the *Add/Remove Programs* dialog (available from the Control Panel).

UPDATE PACKAGE CODE

Package codes are updated automatically or not, according to the selection among the following three options:

- Do not automatically update package code when saving changes

When this radio button is *not* selected, WinINSTALL keeps the same package code when it saves changes to the package. When this radio button *is* selected, WinINSTALL changes the package code whenever it saves changes to the package.

- Automatically update package code when saving changes

When this radio button is selected, WinINSTALL always changes the package code when it saves changes to the package.

- Use global setting {found under *Preferences*}

When this radio button is selected, WinINSTALL will determine whether to update the package code when it saves changes to the package, based on the global setting found under *Preferences*.

***NOTE:** If significant changes are made to a package, the package code should be changed to reflect this. It is not necessary for a package to be changed if the changes to the package are relatively minor.*

GENERAL INFORMATION FOR INSTALLING A WINDOWS INSTALLER FEATURE

Windows Installer packages consist of functional elements called *features* and *components*. *Components* are the smallest elements of an application, as small as a single file, icon, or registry entry. *Features* are the smallest discrete pieces of an application which can be installed. *Features* can contain other *features*, but they must contain one or more *components*. The detailed installation instructions for a *feature* are associated with the *components* that make up that *feature*. However, Windows Installer packages also contain general information about the *features* in the package.

When you highlight a feature in the tree pane and select *General* in the list pane, you can edit this general feature-level information on the following tabs in the data pane:

Summary Tab

The information on this tab includes the feature identifier, the name of the feature as it appears in the console, a description of the feature, and “whether”, “when”, and “where” instructions for the *Add/Remove Programs* applet in Control Panel.

Conditions Tab

The information on this tab includes conditions which will be evaluated to determine whether the feature will be installed.

Advanced Tab

The information on this tab includes the default install levels, where the feature should be stored, whether the feature should be advertised, and how the feature should be displayed.

SUMMARY TAB (WINDOWS INSTALLER FEATURE)

On the *Summary* tab, you can enter, view or edit general information used for management and documentation purposes - such as the name of the feature, its ID, its description, and "whether", "when", and "where" instructions for the *Add/Remove Programs* applet in Control Panel.

***TIP:** A Windows Installer package is made up of one or more features. An example of a feature is the Spell Checker in Microsoft Word.*

On the *Summary* tab in the data pane, you can view the following information:

IDENTIFIER

WinINSTALL automatically generates a unique identifier for each new feature that is created. The identifier has the format `WIFEAT<0000000n>`, where `<0000000n>` is an auto-incremented 8-digit number. WinINSTALL uses this identifier internally to locate features. (WinINSTALL uses a similar format for all internally generated identifiers.) This value is read-only.

NAME

This field holds the name of the feature; this name displays in the tree pane. It is a good idea to use a name that helps users understand what the feature does.

DESCRIPTION

This description of the feature displays as tool-tip text when you mouse-over the feature name in the tree pane of the console.

OTHER FEATURE ATTRIBUTES

- Run from local hard drive

When this radio button is selected, the components of this feature will be installed so that they run from the hard drive of the local machine. The files installed by this feature will be installed to the local hard drive and will be run from the local hard drive.

- Run from source

When this radio button is selected, all of the components of this feature will be installed so that they run from the source media, typically a CD or a network share. The files installed

by this feature will not be installed on the local hard drive, but instead will be run from the network resource or some other installation media.

- Install on first use

When this radio button is selected, the feature will be installed the first time that a user attempts to use it. This is called *just-in-time installation*, or *advertising*.

- Not available

When this radio button is selected, the feature will not display as an installation option.

- Not installable

When this radio button is selected, the feature will display, but will not be an active installation option.

- Other (Use Advanced Tab)

This radio button is selected when you make either of the selections below on the *Advanced* tab:

  - Under Storage Location: *Favor Parent*
  - Under Display: *Hide* Not Available *button*

***NOTE:** Choices made on the radio buttons above can alter what is displayed on the* Advanced *tab, and choices made on that tab can alter what displays here.*

CONDITIONS TAB (WINDOWS INSTALLER FEATURE)

On the *Conditions* tab, you can create, view or modify conditions which will be evaluated in order to determine whether the feature will be installed. Condition definitions are used to set the install level, a numerical value that determines whether a feature is installed and, if so, how it is installed. Conditions allow you to add another level of control to the functionality of an installation.

On the *Conditions* tab in the data pane, you can view or edit the following information:

CONDITION

A condition is an expression that evaluates to either *true* or *false*. You can create condition statements from properties, environment variables, component action states, component install states, feature action states, and feature install states.

LEVEL

Every installation package has a default install level, which is an integer value that can range from 0 to 32,767. Conditions let you change the install-level value for a particular feature, and thus change its installation state. If the install level for a feature is greater than the default install level, the feature will not be installed. An install level of *0* disables and hides the feature.

You can perform the following actions on this tab:

- To create and add a new condition, click the *Add* icon. Enter the required information on the *Feature Condition* dialog and click *OK*.
- To edit an existing condition or install level, highlight it and click the *Edit* icon. Change the desired information on the *Feature Condition* dialog and click *OK*.
- To delete an existing condition, highlight it and click the *Delete* icon.

ADVANCED TAB (WINDOWS INSTALLER FEATURE)

On the *Advanced* tab, you can view, add, or modify information such as install levels, where the feature should be stored, whether the feature should be advertised, and how the feature should be displayed. You can view or edit the following information:

DEFAULT FEATURE INSTALL LEVEL

Packages and features both have default install levels. If the default install level for a feature is less than or equal to the default install level for the package, the feature will be installed. An install level of *0* disables and hides the feature.

DEFAULT PACKAGE INSTALL LEVEL

Every installation package has a default install level, which is an integer value that can range from 0 to 32,767. Features are installed or not installed based on their install level. If the install level for a feature is greater than the default install level, the feature will not be installed.

STORAGE LOCATION

A feature can be installed from the local hard drive or the source media. The following three choices are available:

Favor Local

- First try to install the feature from the local hard drive; if that fails, try to install from the source media.

Favor Source • First try to install the feature from the source media; if that fails, try to install from the local hard drive.

Favor Parent • First try to install the feature using the same installation choice as the parent of the feature; if that fails, try alternative methods.

ADVERTISING

When a feature is advertised, it is not installed until the user tries to use it. This is called *just-in-time installation*, or *advertising,* and it offers certain advantages, such as reducing the network load that accompanies an enterprise-wide simultaneous installation, and not installing applications that a particular user does not need or use. You can choose whether or not to have a feature advertised, and whether to have it advertised based on the degree of integration of the Windows Installer Service with the client operating system. The following advertising choices are available:

Do not allow advertising • Advertising will not be an installation option for this feature.

Favor advertising • Allow the feature to be advertised as an installation option.

Advertise only if the OS understands MSI • Allow the feature to be advertised only if the operating system "understands" how to use the Windows Installer service.

DISPLAY

You can choose to have the feature displayed either expanded or collapsed, or you can choose not to have it displayed at all. You can also hide the *Not available* option from the user. The following display options are available:

Visible and Expanded • The feature will display as an installation option in an expanded form with its components visible.

Visible and Collapsed • The feature will display as an installation option in a collapsed form.

Invisible • The feature will not display as an installation option.

Hide *Not Available* button • When this checkbox is checked, the *Not Available* button does not appear.

NOTE: *Some of the choices that you make on this tab will alter what is displayed on the* Summary *tab, and choices that you make on that tab will alter what displays here.*

GENERAL INFORMATION FOR INSTALLING A WINDOWS INSTALLER COMPONENT

A Windows Installer package installs a product, which can be a single application or a suite of applications. Windows Installer packages are made up of one or more *features* that are in turn made up of one or more *components* and sub-features. A *component* is the smallest and most fundamental unit in a Windows Installer package. In addition to detailed installation instructions about files, registry keys, environment variables, and shortcuts, packages also contain general installation information about components. When you highlight a component in the tree pane and select *General* in the list pane, you can edit information on the following tabs in the data pane:

Summary Tab

The information on the *Summary* tab uniquely identifies the component, tells the installer where it is located on the installation media and where it should be installed on the local machine, specifies the value that will be used to find the component, and optionally defines the conditions under which the component will be installed. The values entered on this tab are used to populate or modify the Components table in the MSI database.

Reserve Cost Tab

The information on this tab designates the amount of disk space (hard drive space in addition to the file size) that will be required if the component is installed. The values entered on this tab are used to populate or modify the ReserveCost table in the MSI database.

Advanced Tab

The information on this tab tells the installer where it should look for the installation files for this component and specifies whether the reference count in the shared DLL registry should be incremented, whether the component should be removed during an uninstall, whether it should be reinstalled if it already exists, and whether an associated condition should be evaluated again during a reinstall. The values entered on this tab are used to populate or modify the Attributes column in the Component table.

If you add new general information about a package, or edit existing information, you must save your changes by clicking the diskette icon on the toolbar or choosing *File --> Save* from the menu bar.

SUMMARY TAB (WINDOWS INSTALLER COMPONENT)

On the *Summary* tab in the data pane, you can enter, view or edit the following information:

IDENTIFIER

This is an internally generated alphanumeric value that uniquely identifies the component for the installer. It is in the format: *WICOMP<0000000n>*, where *<0000000n>* is a unique auto-incremented 8-digit number. This value is used to populate or modify the Component column of the Component table. This value is read-only and is used to relate various tables together.

SOURCE

Specify the private property that represents the folder on the installation media from which the component will be installed. This private property will be resolved at runtime to the location of the MSI file. If you change the existing value, the new entry must begin with *[SourceDir]*\, followed by the path to the location of the component. If the new entry does not start with *[SourceDir]*\, you will receive an error message when you leave this tab.

TARGET

Specify the public property that represents the folder on the local machine to which the component will be installed. This property can be specified in the Property table of the MSI database or on the run line. If no property is specified, it resolves at runtime to the drive with the most free space available. If you change the existing value, you must enter a valid directory property value or click the ellipsis button to browse for one on the *Select Directory* dialog. If you enter an invalid directory property value, WinINSTALL will automatically change it back to *[TARGETDIR]/* when you leave this tab.

COMPONENT

Specify the GUID of the component that will be installed. Click the ellipsis button to create a new GUID.

KEY PATH

Specify the value that will be used by the installer to detect the presence of the component. It can be defined as the location of a specific file or the value of a registry key, or it can be null. If it refers to a file, the file must be present in order for the installer to think that the component is installed. If it is a registry key, that key must be present in order for the installer to think that the component is installed. If it is null, the installer will use the directory as the key path, and if the directory is not present, it will reinstall the component. This value is used to populate or modify the Keypath column of the Component table. Click the ellipsis button to browse for a keypath on the *Define Keypath* dialog.

CONDITION

Create an optional conditional statement that can control whether the component is installed. If the condition evaluates to true or no condition is specified, the component is enabled. If the condition evaluates to false, the component is not installed. This value is used to populate or modify the Condition column of the Component table. Click the ellipsis button to create a condition on the *Condition* dialog.

RESERVE COST TAB (WINDOWS INSTALLER COMPONENT)

On the *Reserve Cost* tab in the data pane, you can enter, view or edit the following information:

RESERVE FOLDER

This is the name of a property that is the full path to the destination directory where the component will be installed. Depending on whether the component is installed to run locally or from source, one of the two amounts of disk space specified below is added to the volume cost of the device containing the destination directory. This value is used to populate or modify the ReserveFolder column of the ReserveCost table.

RESERVE LOCAL

This is the number of bytes of disk space that must be reserved if the component is installed to run locally. If the component is installed to run locally, this amount of disk space specified will be added to the volume cost of the device containing the destination directory. This value is used to populate or modify the ReserveLocal column of the ReserveCost table.

RESERVE SOURCE

This is the number of bytes of disk space that must be reserved if the component is installed to run from source. If the component is installed to run from source, this amount of disk space specified will be added to the volume cost of the device containing the destination directory. This value is used to populate or modify the ReserveSource column of the ReserveCost table.

You can perform the following actions on this tab:

- To reserve space in a new folder, click the *Add* icon. Enter the required information on the *Reserve Cost* dialog and click *OK*.
- To edit existing reserved space, highlight the reserve folder and click the *Edit* icon. change the desired information on the *Reserve Cost* dialog and click *OK*.
- To delete existing reserved space, highlight the reserve folder and click the *Delete* icon.

ADVANCED TAB (WINDOWS INSTALLER COMPONENT)

On the *Advanced* tab in the data pane, you can enter, view or edit the following information:

RUN FROM

Local
- When this radio button is selected, the component cannot be run from source. The entry you make here is used to set or modify the msidbComponentAttributesLocalOnly attribute, and the resulting value is saved in the Attributes column of the Component table.

Source
- When this radio button is selected, the component can be run only from source. The entry you make here is used to set or modify the msidbComponentAttributesSourceOnly attribute, and the resulting value is saved in the Attributes column of the Component table.

Optional
- When this radio button is selected, the component can be run either locally or from source. The entry you make here is used to set or modify the msidbComponentAttributesOptional attribute, and the resulting value is saved in the Attributes column of the Component table.

INCREMENT SHARED DLL COUNTER (CREATE IF NECESSARY)

If this checkbox is checked, the installer will increment the reference count in the shared DLL registry of the component's key path - creating one if it does not yet exist. If this checkbox is *not* checked, the installer will increment the reference count only if the reference count already exists. This entry is used to set or modify the msidbComponentAttributesSharedDllRefCount attribute, and the resulting value is saved in the Attributes column of the Component table.

DISALLOW COMPONENT UNINSTALL DURING PACKAGE REMOVAL

When this checkbox is checked, the installer will not remove the component when a feature or application that includes the component is uninstalled. The installer registers an extra system client for the component in the Windows Installer registry settings. The entry you make here is used to set or modify the msidbComponentAttributesPermanent attribute, and the resulting value is saved in the Attributes column of the Component table.

SKIP COMPONENT INSTALLATION WHEN KEY PATH EXISTS

When this checkbox is checked, the installer does not install or reinstall the component if a key path file or a key path registry entry for the component already exists. The application does register itself as a client of the component. The entry you make here is used to set or modify the msidbComponentAttributesNeverOverwrite attribute, and the resulting value is saved in the Attributes column of the Component table.

RE-EVALUATE COMPONENT CONDITION DURING REINSTALL

When this checkbox is checked, the installer evaluates any conditions associated with the component whenever the component is reinstalled. If the condition previously evaluated to *False* and now evaluates to *True*, the installer installs the component. If the condition previously evaluated to *True* and now evaluates to *False*, the installer removes the component even if the component has other products as clients. The entry you make here is used to set or modify the msidbComponentAttributesTransitive attribute, and the resulting value is saved in the Attributes column of the Component table.

12

ADDITIONAL MSI PACKAGE OPTIONS

When you highlight a package in the tree pane and select *General* in the list pane, different tabs and sub-tabs appear in the data pane, depending on what you have selected in the tree pane. Some of the tabs in the data pane are specific to features or components of Windows Installer packages. This chapter discusses two tabs that display only for Windows Installer packages.

When you highlight a Windows Installer package in the tree pane and select *General* in the list pane, the following tabs and sub-tabs display in the data pane.

- Directory Tab
- Install Modes Tab
- Advanced Tab
  - Components Sub-tab
  - Launch Condition Sub-tab
  - Sequence Sub-tab
  - Install Sub-tab
  - Admin Sub-tab
  - Advertise Sub-tab
  - AppSearch Sub-tab
  - Dialogs Sub-tab
  - Custom Actions Sub-tab
  - Package History Sub-tab

NOTE: *Windows Installer packages, features, and components all have an Advanced tab, but each has different information. The Advanced tabs for packages are covered in this chapter. Those for features and components are explained in the Advanced Tab (Windows Installer Feature) and Advanced Tab (Windows Installer Component) sections of the General Package Options chapter of this guide.*

This chapter also provides a brief discussion of the WinINSTALL MSI Table Editor.

DIRECTORY TAB

You can view, add, edit or delete the directories used in a Windows Installer package on the *Directory* tab in the data pane. Each directory may be associated with a component, and if so, also associated with the feature which contains that component.

When you highlight a Windows Installer package in the tree pane, select *General* in the list pane, and click the *Directory* tab in the data pane, you can enter, view or edit all of the directories in the package.

On the *Directory* tab, you can enter or modify the following information: *Directory ID*, and

Directory Name

TO MANAGE DIRECTORIES USED IN A WINDOWS INSTALLER PACKAGE:

1. **Navigate to the package under *Software Distribution Lists* in the tree pane.**
2. **Highlight *General* in the list pane.**
3. **Click the *Directories* tab in the data pane. You can view or edit the following information:**
   - **Directory ID**
   - **Directory Name**
4. **You can perform the following action on this tab:**
   - **To add a directory, click the *New* icon. Enter the required information on the Directory dialog and click *OK*.**
   - **To modify an existing directory, highlight it and click the *Edit* icon. Edit the desired information on the Directory dialog and click *OK*.**

INSTALL MODES TAB

You can edit Windows Installer packages to specify whether to install the application per user or per machine, whether to reboot after installation, and what type of user interface to use. Per-machine installations are installed once for the system; per user installations are performed for individual users on a system, which could mean multiple installations of the same application on a system.

TO SEE OR ENTER INSTALL MODE INFORMATION FOR A PACKAGE:

1. **Navigate to the package under *Software Distribution Lists* in the tree pane.**
2. **Highlight *General* in the list pane.**

3 **Click the *Install Modes* tab in the data pane.**

4 **You can edit the Installation Mode by selecting one of the following radio buttons, depending on the desired behavior:**

- **Install only per user**

Installation should be performed based on the user who is currently logged on to the PC where the installation is being performed.

- **Install only per machine user**

Installation should be performed based on the PC where the installation is being performed.

- **Install per machine; if fails, per user**

Windows Installer should first attempt to perform the installation based on the PC where the installation is being performed. If installation is not possible, Windows Installer should than attempt to perform the installation based on the user who is currently logged on.

5 **To automatically reboot the user's PC after the application installation is complete, select the check box marked *Reboot the workstation when this application installation is complete*.**

6 **To allow the user to see only progress dialogs and error messages while the installation is performed, select the check box marked *Show basic user interface only (simple progress and error handling)*.**

7 **To save your entries, click the *Save* icon or choose *Save* from the *File* menu.**

ADVANCED TAB

On the *Advanced* tab, you can enter, view or edit installation information for a Windows Installer package, including the components within it, the conditions that must be met in order for the product to be installed, the sequence of actions that will take place during a regular install, an administrative install, and an advertised install, and how the installer will locate files. This information is presented on six sub-tabs:

- Components sub-tab.
- Launch Condition sub-tab (see the Windows Installer Launch Conditions section of the Package Conditions chapter, later in this guide).
- Sequence sub-tab.
- AppSearch sub-tab.
- Dialogs sub-tab.

- Custom Actions sub-tab.

COMPONENTS SUB-TAB

Windows Installer packages are made up of *features*, which are, in turn, made up of *components* or other *features*. *Components* hold various files, shortcuts, and/or registry entries. On the *Components* tab, you cannot create a component directly, but you can view the following information for existing components:

- Component

This is an internally generated alphanumeric value that uniquely identifies the component to WinINSTALL. It is in the format: *WICOMP<0000000n>*, where `0000000n` is a unique auto-incremented 8-digit number.

All internally generated WinINSTALL identifiers have the same format: *WI + <entity identifier> + <0000000n>*.

Entity identifiers include *COMP* (for components), *FEAT* (for features), etc.

- GUID

A *Globally Unique Identifier* (GUID) is a 128-bit unique identification string used to identify different products, upgrades, and installer components. Each installer component used by Windows Installer must have a unique GUID, and every version of every product must have a unique package GUID. Different upgrades of a product, however, have the same product GUID, though each will have a unique upgrade GUID.

An example of a syntactically correct GUID is shown below:

```
97531-2222-44444-666666-88888888ABCD
```

- Key Path

The key path defines the location of the component on the system. This is normally the name of a file in the component, but it can also be a folder or a registry entry. The Windows Installer looks for the key path of a component to determine if the component is properly installed.

- Key Path Value

A key path value is uniquely associated with a component - a component can have only one key path value and a key path value can be used by only one component. The Windows Installer uses this value to locate an installed component and also uses it for self-repair if it finds that the key path is missing.

LAUNCH CONDITION SUB-TAB

The Launch Condition sub-tab is described in detail in the Windows Installer Launch Conditions section of the Package Conditions chapter, later in this guide.

SEQUENCE SUB-TAB

There are three types of installations for Windows Installer packages - a normal installation, an administrative installation, and an advertised installation. Each type of installation has two different sequences of actions - one for the user interface and one for the actual installation of the product. Within each sequence, each separate action has a name, an optional condition, and a sequence number that determines the order of execution.

After you select the *Sequence* tab in the data pane, you must click one of the following sub-tabs:

- Install sub-tab
- Admin sub-tab
- Advertise sub-tab

On the sub-tab you select, you must click an additional sub-tab:

- Execute sub-tab
- User Interface sub-tab

INSTALL SUB-TAB (EXECUTE AND USER INTERFACE SUB-TABS)

On the *Execute* sub-tab of the *Install* sub-tab, you can view, add, modify or delete information related to the sequence of actions performed during a normal installation. The installation sequence is required.

On the *User Interface* sub-tab of the *Install* sub-tab, you can view, add, modify or delete information related to the sequence of UI actions performed during a normal installation. The UI sequence is optional, depending on whether or not the installation requires a user interface.

NOTE: *The INSTALL action is a top-level action called to install or remove components. This action queries the InstallUISequence Table and InstallExecuteSequence Table for the action to execute, the condition for action execution, and the place of the action in the sequence.*

You can view or edit three items on this tab:

- Action
- Condition
- Sequence Number

You can perform the following actions on this tab:

- To add a new sequence entry to the installation, click the *Add* icon. Enter the required information on the *Sequence Entry* dialog and click *OK*.
- To edit the action, condition, or sequence number in an existing sequence entry, highlight it and click the *Edit* icon. Make the desired changes on the *Sequence Entry* dialog and click *OK*.
- To delete an existing sequence entry, highlight it and click the *Delete* icon.

ADMIN SUB-TAB (EXECUTE AND USER INTERFACE SUB-TABS)

An administrative installation installs a source image of the application onto the network to make it available to network users for local installations. The output of an administrative installation is analogous to a source image on a CD-ROM. Users in a workgroup who have access to this administrative image can then install the product from this source.

***TIP:** Editing an administrative installation is usually a concern of the original software vendor, rather than a network administrator at a customer site.*

On the *Execute* sub-tab of the *Admin* sub-tab, you can view, add, modify or delete information related to the sequence of actions performed during an administrative installation. The installation sequence is required.

On the *User Interface* sub-tab of the *Admin* sub-tab, you can view, add, modify or delete information related to the sequence of UI actions performed during an administrative installation. The UI sequence is optional, depending on whether the installation requires a user interface.

You can view or edit the following information on this tab:

- Action
- Condition

- Sequence Number

You can perform the following actions on this tab:

- To add a new sequence entry to the installation, click the *Add* icon. Enter the required information on the *Sequence Entry* dialog and click *OK*.
- To edit the action, condition, or sequence number in an existing sequence entry, highlight it and click the *Edit* icon. Make the desired changes on the *Sequence Entry* dialog and click *OK*.
- To delete an existing sequence entry, highlight it and click the *Delete* icon.

ADVERTISE SUB-TAB (EXECUTE AND USER INTERFACE SUB-TABS)

Advertising is a technique whereby an application appears to be installed, but is not actually installed until the user attempts to use the application. This is also called a *just-in-time installation*.

On the *Execute* sub-tab of the ***Advertise*** sub-tab, you can view, add, modify or delete information related to the sequence of actions performed during an advertised installation. The installation sequence is required.

On the *User Interface* sub-tab of the *Advertise* sub-tab, you can view, add, modify or delete information related to the sequence of UI actions performed during an advertised installation. The UI sequence is optional, depending on whether the installation requires a user interface.

You can view or edit the following information on this tab:

- Action
- Condition
- Sequence Number

You can perform the following actions on this tab:

- To add a new sequence entry to the installation, click the *Add* icon. Enter the required information on the *Sequence Entry* dialog and click *OK*.
- To edit the action, condition, or sequence number in an existing sequence entry, highlight it and click the *Edit* icon. Make the desired changes on the *Sequence Entry* dialog and click *OK*.
- To delete an existing sequence entry, highlight it and click the *Delete* icon.

APPSEARCH SUB-TAB

The information displayed on the *AppSearch* tab tells the installer what to look for to determine where an application should be installed. You can specify the signature for which the installer should search and also indicate where to look. For example, you can tell the installer to look for locations specified in the registry, or the location of an existing file, or something else, any of which can be used to populate a variable that will then determine the location where the application will be installed. The Windows Installer uses this information to locate the components of an application so that it can verify that all of the required components exist and can be found before performing an installation and to perform self-repair if a component is missing.

You can view or edit the following information on the *AppSearch* sub-tab:

Signature • A name associated with the search. A list of file signatures is maintained in the Signature table in the MSI database. At least one of these files needs to be present on a user's computer for the user to be in compliance with the program. (The signature is also the external key into the Signature, RegLocator, IniLocator, CompLocator, and DrLocator tables in the MSI database.)

Property • The property which AppSearch will set to the results of the search - for example, the location of the file indicated by the signature specified above. The AppSearch action sets this property if the file signature exists on the user's computer. If the signature is not found, the initial property value is used. If the signature is not found and the property was never initialized, the property is undefined. This must be a public property and have an identifier that contains no lowercase letters.

CCP • Compliance Checking Program (CCP) - uses file signatures to make sure that the user is in compliance with the program.

RMCCP • The RMCCPSearch action uses file signatures to validate that qualifying products are installed on a system before an upgrade installation is performed. (Each file signature in the CCPSearch table is searched for under the path referred to by the CCP_DRIVE property using the DrLocator table. The absence of the signature from the Signature table denotes a directory. If any signature is determined to exist, the RMCCPSearch action terminates.)

You can perform the following actions on the AppSearch sub-tab:

- To add a new signature, click the *Add* icon. Enter the required information on the *AppSearch and Related Tables* dialog and click *OK*.

- To modify an existing signature, highlight it and click the *Edit* icon. Make the desired changes on the *AppSearch and Related Tables* dialog and click *OK*.
- To delete an existing signature, highlight the signature and click the *Delete* icon.

DIALOGS SUB-TAB

You can edit Windows Installer packages to create dialogs that your client machines can use when the package is processed.

ADDING DIALOGS

You can perform the following actions on the *Dialogs* sub-tab, with regard to adding dialogs to a Windows Installer package:

- To add a new dialog, click the *Add* icon. Enter a name for the dialog on the *New Dialog* dialog and click *OK*. Design the dialog box on the *Dialog*: dialog and the *Options* dialog.
- To edit an existing dialog, highlight it and click the *Edit* icon. Make the desired changes on the *Dialog*: dialog and the *Options* dialog.
- To delete an existing dialog, highlight it and click the *Delete* icon.

SIMPLE GUI

WinINSTALL also provides the ability to quickly and easily generate a standard Windows Installer user interface, including a customized dialog to prompt the user during an install for a value to assign to a property. This property prompt dialog is based on a template MSI file, *QueryProperty.msi*, found in the *\bin* directory of the WinINSTALL share.

To access the simple GUI/user prompt feature, click the *Simple GUI* button on the Dialogs tab. The *Dialog Properties* dialog will appear, allowing you to add as many property prompt dialogs as you like.

This dialog allows you to set the name of the property, a description to appear at the top of the page when it's shown to the user during an install, and a prompt to appear just above the text box provided for the user's input.

When the *Dialog Properties* dialog appears, it lists all existing user prompt dialogs by property in the *Properties* list box. Selecting a listed property will display the existing values for that property's prompt dialog *Page Description* and *Prompt* (see below).

To add a user prompt dialog, click the *Add* button. The *Dialog Property* dialog will appear, allowing you to enter the following three items:

Property • The property whose value the user will be prompted to enter. WinINSTALL will write the text provided by the user to this property before the install is performed (for example, *[TARGETDIR]*).

Page Description • The title of the user prompt dialog. This title will display in the title bar of the dialog at install time, when the user is prompted (for example, *Data Directory*).

Prompt • The text to display on the dialog to prompt for the desired information (for example, *Please enter the data directory:*)

Generally, you will specify a public MSI property in the *Property* field (a name that is in all caps), but any valid property is acceptable. During package installation, the property's value is set to the user's input prior to the *InstallExecute* sequence of the install.

***NOTE:** When you Click OK on this dialog and then save the MSI file, WinINSTALL generates a number of user interface dialogs for the MSI file. You can edit these as desired.*

To modify a user prompt dialog, select the corresponding property in the list and click the *Modify* button. The *Dialog Property* dialog will appear, allowing you to modify the *Property*, *Page Description*, and *Prompt*.

To delete a user prompt dialog, select the corresponding property in the list and click the *Delete* button.

HOW TO MODIFY THE APPEARANCE OF THE SIMPLE GUI DIALOGS

You can modify the template MSI file, *QueryProperty.msi*, as desired, for example to change the images in the Binaries table to use your company logo.

Alternatively, WinINSTALL provides the option of specifying a different template MSI file to use. Click on the *Advanced Options* button on the *Dialog Properties* dialog to view the *Advanced Dialog Properties* dialog.

By default, the *Advanced Dialog Properties* dialog will display *QueryProperty.msi*, the default template MSI file included with WinINSTALL. But you can enter or browse to another .msi file.

WARNING: *WinINSTALL does not check to verify that the template MSI file is appropriate. You will need to make sure that it is suitable to replace the supplied* QueryProperty.msi *file.*

CUSTOM ACTIONS SUB-TAB

You can edit MSI packages to create custom actions that will occur when the package is processed. The Custom Action Wizard, which helps you edit or create MSI custom actions, is discussed later in this chapter, and addtional details are also presented later in this chapter, in the section entitled MSI Custom Action Example.

You can view or edit the following information on the Custom Actions sub-tab:

Name	•	Descriptive label for the custom action
Type	•	Source type (EXE, DLL, Install, JScript, Text, VBScript)
Source	•	Location where the source of the custom action is stored
Target	•	Target of the custom action (differs depending on source type)

You can perform the following actions on this tab:

- To add a custom action, click the *Add* icon. The Custom Action Wizard will walk you through the creation of a custom action.
- To modify an existing custom action, highlight it and click the *Edit* icon. The Custom Action Wizard will step you through the process of modifying the custom action.
- To delete an existing custom action, highlight it and click the *Delete* icon.

PACKAGE HISTORY

If *Always log changes* is selected from the drop-down on the Package History sub-tab, WinINSTALL will record in the MSI file itself a history of modifications made to the selected package within the WinINSTALL Console.

The drop-down provides three choices: *Always log changes*, *Never log changes*, or *Log according to console defaults*.

NOTE: *To set the console defaults for the* MSI Package History *Feature, select any Windows Installer package in the Console tree pane and then select* Edit->Preferences. *On the* Preferences *dialog, check or uncheck* Log MSI changes to the changed MSI package *and click* OK.

On the *Package History* sub-tab, you can view the following information for each logged change:

- date and time of the change
- type of change
- associated message
- username of the user who was logged into the Console machine at the time of the change.

You can filter the list to view only certain types of changes, you can purge the list by date, and you can export the log to a tab-delimited text file.

***NOTE:** Package History records only three types of package events: changes to MSI tables, merging of Merge Modules, and application of Transforms.*

***WARNING:** Changes made outside the WinINSTALL Console, as well as those changes to packages made in the WinINSTALL table editor or through MSI Validation and Repair, are not logged to the Package History table.*

THE CUSTOM ACTION WIZARD

The *Custom Action Wizard* creates or modifies a custom action that can be used to extend the built-in functionality of the Windows Installer. Once you have created a custom action, you must insert it into one of the sequences in order for it to be executed. Once the custom action is defined, it can be used like any other action in the sequence table. Additional details on defining custom actions and adding them to the sequence table are explained in the section entitled MSI Custom Action Example, below.

To invoke the *Custom Action Wizard*, select the desired Windows Installer package in the Console tree view, select *General* in the list view, and then click on the *Advanced* tab and the *Custom Actions* sub-tab in the data pane. This tab presents a list of all configured custom actions for the selected package.

To create a new custom action, click the *Add* icon; to edit an existing custom action, select the desired custom action and click the *Edit* icon. Either icon will launch the *Custom Action Wizard*.

After presenting a *Welcome* panel explaining its purpose, the Custom Action Wizard gathers information from you on the four panels described below. After the last, a *Completing the Custom Action Wizard* announces the completion of the process, and clicking *Finish* will actually create the custom action.

SOURCE TYPE

On the *Source Type* panel, you provide a name for the custom action (alphanumeric characters only--no spaces or special characters) and select the source type for the custom action to be created. Custom actions can be any of six source types:

- **DLL** • Calls an entry point into a DLL.
- **EXE** • Launches an executable.
- **Text** • Defines a directory, property or error message with specified text.
- **JScript** • Launches a Java script.
- **VBScript** • Launches a VBScript script.
- **Install** • Launches a nested installation of another MSI package.

SOURCE STORAGE

On the *Source Storage* panel, select the appropriate radio button to specify where in the package the source of the custom action should be stored. The available radio buttons depend on the source type selected on the previous panel:

- **DLL** • *Binary Table* or *File Table*
- **EXE** • *Binary Table*, *File Table*, *Directory*, or *Property*
- **Text** • *Error Text*, *Directory*, or *Property*
- **JScript** • *Binary Table*, *File Table*, *Line of Code*, or *Property*
- **VBScript** • *Binary Table*, *File Table*, *Line of Code*, or *Property*
- **Install** • *Binary Table*, *Relative Path to MSI*, or *Product Code*

The choice made on this that you select determines which target panel will appear next.

TARGET PANELS

Depending on the source storage choice made on the previous panel, one of the following target panels will appear.

BINARY ENTRY PANEL

If *Binary Table* was selected on the *Source Storage* panel, the *Binary Entry* panel appears, displaying a list of binary files included in the package. You can add an additional binary file by clicking the *Add Binary Entry* button and then entering or browsing to the desired file. Select the desired file in the list and click *Next*.

FILE PANEL

If *File Table* was selected on the *Source Storage* panel, the *File* panel appears, displaying a list of files included in the package. You can add an additional file by clicking the *Add File Entry* button and then entering or browsing to the desired file. Select the desired file in the list and click *Next*.

If *Relative Path to MSI* was selected on the *Source Storage* panel, the *File* panel appears, displaying a list of MSI file entries. You can add an additional MSI file by clicking the *Select Relative MSI . . .* button and then entering or browsing to the desired MSI file. Select the desired file in the list and click *Next*.

DIRECTORY PANEL

If *Directory* was selected on the *Source Storage* panel, the *Directory* panel appears, prompting you to enter or browse to the directory containing the custom action. Click *Next* to proceed.

PROPERTY PANEL

If *Property* was selected on the *Source Storage* panel, the *Property* panel appears, displaying a list of available Windows Installer properties. Select the desired property and and click *Next*.

TARGET PANEL

If *Error Text* was selected on the *Source Storage* panel, the *Target* panel appears, displaying an edit box where you can enter a formatted text string, a literal message, or an index into the MSI file error table. Type in the desired entry and click *Next*.

If *Line of Code* was selected on the *Source Storage* panel, the *Target* panel appears, displaying an edit box where you can enter a line of the appropriate code (JScript or VBScript). Type in the desired code and click *Next*.

PRODUCT CODE PANEL

If *Product Code* was selected on the *Source Storage* panel, the *Product Code* panel appears, prompting you to enter the GUID for the MSI package to install as a custom action. Enter the GUID and click *Next* to proceed.

FINAL DETAILS

On the *Final Details* panel, you provide instructions on three aspects of the actual execution of the custom action.

EXECUTION OF CUSTOM ACTION

Always • Always execute. Because a custom action can be listed in both the UI and the Execute Sequence tables, it could potentially be executed twice if *Always* is selected.

First Sequence • Execute once if present in both sequence tables.

Once Per Process • Execute once if present in both sequence tables. Used to prevent actions that modify the session state, such as property and database data, from running twice.

Client Repeat • The action runs only if the execute sequence is run on the client following UI sequence. May be used to provide either/or logic, or to suppress the UI related processing if already done for client session.

Inscript • Specifies a deferred execution custom action (queues the custom action for execution at scheduled point within script).

Rollback checkbox • Available only for deferred execution (Inscript) custom actions, designates the custom action as one to be performed only on install roll back.

Commit checkbox • Available only for deferred execution (Inscript) custom actions, designates the custom action as one to be performed only on install commit.

RETURN TYPE

The *Return Type* depends on the state of the *Continue* and *Asynchronous* checkboxes, as explained below.

NOTE: *The* Asynchronous *checkbox is ignored if the* Rollback *checkbox (see Execution of Custom Action, above) is checked, and it is disabled if the custom action type is* Install *(nested MSI execution--see Source Type, above).*

BOTH CONTINUE AND ASYNCHRONOUS UNCHECKED

If neither is checked, the installer waits for the custom action thread to complete before resuming the main installation thread, and if the custom action thread does not return a successful return code (i.e., 0), the installation fails.

CONTINUE CHECKED

If *Continue* is checked but *Asynchronous* is not checked, the installer waits for the custom action thread to complete before resuming the main installation thread, but the return code is ignored, and the installation continues in any case.

ASYNCHRONOUS CHECKED

If *Continue* is not checked but *Asynchronous* is checked, the installer runs the custom action simultaneously as the main installation continues, but the return code is checked at the end of the sequence. If the return code does not signal success (i.e., 0), the install fails.

BOTH CONTINUE AND ASYNCHRONOUS CHECKED

If both are checked, the installer runs the custom action simultaneously as the main installation continues, but the return code is ignored at the end of the sequence.

DO NOT IMPERSONATE CHECKBOX

The installer runs custom actions with user privileges by default for security reasons. If this checkbox is checked, then the installer will run with only user level privileges.

This checkbox is only available for deferred execution (Inscript) custom actions (see Execution of Custom Action, above).

***TIP:** The state of the* Do Not Impersonate *checkbox is not an issue if the installer is being launched by the WinINSTALL agent, since the WinINSTALL agent normally executes with elevated privileges.*

COMPLETING THE WININSTALL CUSTOM ACTION WIZARD

On this panel, you tell the wizard to complete the process and actually create the custom action.

MSI CUSTOM ACTION EXAMPLE

The following section uses the simple launching of an external program as an example of a Windows Installer custom action. Such actions can be configured to executed as part of a package installation or uninstallation, and at any point during the install or uninstall process.

The example includes four variations, demonstrating how to call an external program at the start or completion of the installation or uninstallation of a package. These four variations are intended to serve as a first step in understanding how to create and configure custom actions for Windows Installer packages.

Configuring any of these simple behaviors is straightforward, and involves two separate actions. First you must set up the custom action itself. Once the custom action has been created, you must then add it to the sequence table, which includes specifying at what point in the process and under what conditions it will be executed. WinINSTALL makes these actions very simple and easy to perform.

***TIP:** For additional details on custom actions and their configuration, please see the Windows Installer help file,* MSI.CHM*, available from Microsoft.*

DEFINING THE CUSTOM ACTION:

You can define an external program as a custom action within an MSI package by running the WinINSTALL Custom Action Wizard:

1. **Select the desired Windows Installer package in the tree pane, select *General* in the list pane, select the *Advanced* tab in the data pane and select the *Custom Actions* sub-tab.**
2. **Click the *Add* icon to launch the Custom Action Wizard.**
3. **On the *Source Type* panel, provide a name for the custom action (alphanumeric characters only--no spaces or special characters), select *EXE* as the type, and click *Next*.**
4. **On the *Source Storage* panel, select the *Directory* radio button, and click *Next*.**
5. **When the *Directory* panel appears, leave the *Enter directory where the custom action may be found* field blank, and click *Next*.**
6. **On the *Target* panel, enter the UNC path and filename of the target executable, along with any command line parameters. Note that properties can be included here (e.g. *[SourceDir]winstala.exe*).**

***NOTE:** If you specify the* SourceDir *property, you must also add the* ResolveSource *action to the sequence table prior to the custom action.*

Be sure to include quotation marks around the executable path if you are using long path names.

Click *Next*.

7 **On the *Final Details* panel, select *Inscript/Commit* for the *Execution of custom action* and *Continue* as the *Return Type*.**

8 **The *Completing* panel reminds you that you will need to add the custom action to the sequence table in the appropriate place (see the additional steps below). Click *Finish* to complete the wizard, and then click the *Save* icon (diskette) on the tool bar to save the package.**

9 **To complete the preparation of your custom action, you will need to specify where in the install sequence the custom action should be executed, as well as the appropriate sequence condition. These specifications differ, depending on your intent, and are detailed below for pre-install, post-install, pre-uninstall, and post-uninstall situations.**

SPECIFYING THE CUSTOM ACTION SEQUENCE:

To instruct the Windows Installer to execute your custom action at the appropriate time and under the appropriate conditions, you will need to follow one of the procedures outlined below, depending on your intentions.

PRE-PROCESSING DURING INSTALL:

To enter your custom action to execute at the start of the package installation, you must add it to the sequence table immediately following the *InstallInitialize* action by following these steps:

1 **Select the *Advanced* tab, the *Sequence* sub tab, the *Install* sub tab, and the *Execute* sub tab, then sort the events by sequence.**

2 **Choose a sequence number between the *InstallInitialize* event and the event immediately following it.**

3 **Click the *Add* icon to add your custom action into the sequence.**

4 **On the *Sequence Entry* dialog, enter the sequence number you have chosen, type in *NOT Installed* as the sequence condition, and click the *Ellipsis* button beside the *Action* field to browse for and select your custom action.**

5 **Click *OK* to add the custom action to the sequence and again save the package.**

POST-PROCESSING DURING INSTALL:

To enter your custom action to execute at the end of the package installation, you must add it to the sequence table immediately before the *InstallFinalize* action by following these steps:

1. **Select the *Advanced* tab, the *Sequence* sub tab, the *Install* sub tab, and the *Execute* sub tab, then sort the events by sequence.**
2. **Choose a sequence number between the *InstallFinalize* event and the event immediately preceding it.**
3. **Click the *Add* icon to add your custom action into the sequence.**
4. **On the *Sequence Entry* dialog, enter the sequence number you have chosen, type in *NOT Installed* as the sequence condition, and click the *Ellipsis* button beside the *Action* field to browse for and select your custom action.**
5. **Click *OK* to add the custom action to the sequence and again save the package.**

PRE-PROCESSING DURING UNINSTALL:

To enter your custom action to execute at the start of the package uninstallation, you must add it to the sequence table immediately following the *InstallInitialize* action by following these steps:

1. **Select the *Advanced* tab, the *Sequence* sub tab, the *Install* sub tab, and the *Execute* sub tab, then sort the events by sequence.**
2. **Choose a sequence number between the *InstallInitialize* event and the event immediately following it.**
3. **Click the *Add* icon to add your custom action into the sequence.**
4. **On the *Sequence Entry* dialog, enter the sequence number you have chosen, type in *Installed* as the sequence condition, and click the *Ellipsis* button beside the *Action* field to browse for and select your custom action.**
5. **Click *OK* to add the custom action to the sequence and again save the package.**

POST-PROCESSING DURING UNINSTALL:

To enter your custom action to execute at the end of the package uninstallation, you must add it to the sequence table immediately before the *InstallFinalize* action by following these steps:

1. **Select the *Advanced* tab, the *Sequence* sub tab, the *Install* sub tab, and the *Execute* sub tab, then sort the events by sequence.**
2. **Choose a sequence number between the *InstallFinalize* event and the event immediately preceding it.**

3. **Click the *Add* icon to add your custom action into the sequence.**
4. **On the *Sequence Entry* dialog, enter the sequence number you have chosen, type in *Installed* as the sequence condition, and click the *Ellipsis* button beside the *Action* field to browse for and select your custom action.**
5. **Click *OK* to add the custom action to the sequence and again save the package.**

THE MSI TABLE EDITOR

If you right-click on a Windows Installer package in the tree view, the context menu presents a choice labeled *View/Edit Tables*. Making this selection will display the WinINSTALL MSI Table Editor, a facility for viewing and editing the full contents of any Windows Installer package.

This editor is also available from the *View Tables* button on the *Advanced Options* panel of the WinINSTALL Patch Wizard. In this situation, the Table Editor displays the intermediate patch creation package (.pcp) file being used to create the patch. Edits entered at this point will be included in the resulting .msp file.

***WARNING:** The WinINSTALL Table Editor does not assure that your changes will not break the package. Because only minimal validation is enforced in the Table Editor, it is strongly recommended that you make changes elsewhere in the Console, unless you have a specific need to use the Table Editor. In any case, it is critical that you understand the possible implications of any changes you make in the Table Editor.*

The left column of the Table Editor lists, in alphabetical order, the tables contained in the package. On the right is a grid displaying the rows and columns of the table selected in the list on the left, with values for each field visible and available for edit.

HOW TO CHANGE VALUES IN THE MSI TABLE EDITOR

To edit a value, click on desired cell and enter the desired changes. Note that validation is minimal--you are not prevented from entering invalid entries or entries which will invalidate other, linked values in other tables.

To add or remove rows, take either of these actions:

1. **Select *Add Row* or *Drop Row(s)* from the Edit menu.**
2. **Right-click in the grid and select *Add Row* or *Drop Row(s)*.**

HOW TO SAVE OR DISCARD CHANGES IN THE MSI TABLE EDITOR

When you have finished your edits, you can choose one of two methods to either save your changes or close the editor without saving your changes.

1. **Select *Save* or *Close* from the *File* menu.**
2. **Click the *Save* button or the *Close* button at the bottom of the Table Editor dialog.**

13

FILES

One of the most important tasks performed during a package installation is the modification of files on the target system – adding files, removing files, moving files, copying files, and creating folders, including decisions about what to do when destination files already exist on the target system.

When you highlight a package in the tree pane of the Console and select *Files* in the list pane, you can use the tabs in the data pane to examine or change the package's instructions regarding files. You can specify the following file operations:

- Files to be added.
- Folders to be created.
- Files to be removed, and under what circumstances.
- Files to be moved.
- Files to be duplicated.
- ODBC information to be added or modified.

WINDOWS INSTALLER PACKAGE FILE OPERATIONS

When you highlight a Windows Installer package and select *Files* in the list pane, six tabs in the data pane are available for controlling file operations: *Add*, *Remove*, *Move File*, *Duplicate File*, *Create Folder*, and *ODBC*.

You can specify a number of file modifications to be made on the target system when a Windows Installer package, feature, or component is processed – files to be added, removed, moved, or copied; folders to be created: and changes to be made to ODBC data source, driver, and translator files.

To provide instructions for file modifications, select a Windows Installer package in the tree pane, highlight *Files* in the list pane, and click one of the six tabs in the data pane.

ADD FILES

The *Add* tab presents a list of files that will be added to the target system during installation of a Windows Installer package, feature, or component. The following information displays for each file in the list:

LONG FILENAME

This is the long version of the name of the file that will be added to the target system during installation. This is a text string.

SHORT FILENAME

This is the short version of the name of the file that will be added to the target system during installation. This is a text string that uses short file name syntax - eight-character name, period (.), and 3-character extension. A short file name must always be provided because the SHORTFILENAMES property may be set or the target volume for the installation may only support short file names.

SIZE

This is the size, in bytes, of the file to be added to the target system during installation.

VERSION

This is the version of the selected component file that will be added to the target system during installation.

ATTRIBUTES

These are the file attributes - such as *Read-Only*, *Hidden*, or *System* - of the file that will be added to the target system during installation.

LANGUAGE

This is the language identified with the file that is to be added to the target system during installation.

TARGET DIRECTORY

This is the full path to the directory to which the file will be added on the target system during installation. (The default value provided by WinINSTALL is the variable *[TARGETDIR]*\.)

You can view, add, modify or delete information about files that will be added to the target system during installation:

- To add a new file to the list of files to be added during installation, click the *New* icon. If you highlighted a package or a feature in the tree pane, you will be prompted to select a specific component within that package or feature before continuing. Navigate to the desired file on the *Files to Insert* dialog and click *OK*.

- To modify an existing file on the list of files to be added during installation, highlight the file and click the *Edit* icon. Make the desired changes on the *File Properties* dialog and click ***OK***.
- To delete a file from the list of files to be added during installation, highlight the file and click the *Delete* icon.

REMOVE FILES

The *Remove* tab presents a list of files that will be removed from the target system during installation of a Windows Installer package, feature, or component, and the circumstances under which they are to be removed. The following information displays for each file in the list:

FILE NAME

This is the name of a file that is to be removed from the target system when the MSI package is processed.

MODE

This specifies the circumstances under which the file is to be removed from the target system:

- during installation of the component
- during removal of the component
- during installation or removal of the component

You can perform the following actions on this tab:

- To add a new file to the list of files to be removed during installation, click the *New* icon. If you highlighted a package or a feature in the tree pane, you will be prompted to select a specific component within that package or feature before continuing. Navigate to the desired file on the *File(s) to Remove* dialog and click *OK*.
- To set or change settings for when a file will be removed during installation, highlight the file and click the *Edit* icon. Navigate to a different file on the *Remove Properties* dialog and click *OK*.
- To delete a file from the list of files to be removed during installation, highlight the file and click the *Delete* icon.

MOVE FILES

The *Move* tab presents a list of files that will be moved from one location on the target system to another during installation of a Windows Installer package, feature, or

component. (This tab lets you view or modify information relating to the MoveFile table.) The following information displays for each file in the list:

SOURCE NAME

This is the original name of the file that will be moved from one location to another on the target system during installation.

DEST NAME

This is the name of the file after it is moved from one location to another on the target system during installation.

SOURCE FOLDER

This is the original full-path directory location of the file that will be moved from one location to another on the target system during installation.

DEST FOLDER

This is the new full-path directory location to which the file will be moved on the target system during installation.

You can perform the following actions on this tab:

- To add a new file to the list of files to be moved during installation, click the *New* icon. If you highlighted a package or a feature in the tree pane, you will be prompted to select a specific component within that package or feature before continuing. Enter the required information on the *Move File* dialog and click *OK*.
- To modify an existing file on the list of files to be moved during installation, highlight the file and click the *Edit* icon. Make the desired changes on the *Move File* dialog and click *OK*.
- To delete a file from the list of files to be moved during installation, highlight the file and click the *Delete* icon.

DUPLICATE FILES

The *Duplicate File* tab presents a list of installed files (selected from among the files to be added) to be duplicated on the target system to another location during installation of a Windows Installer package, feature, or component. The aim of this function is to allow a single file to be installed to multiple locations.

Information entered on this tab affects the contents of the DuplicateFile table.

The following information appears for each file in the list:

FILE

This is the name of the file being added during installation that will also be duplicated.

DESTINATION NAME

This is the name of the duplicated file when it is created on the target system.

DESTINATION FOLDER

This is the folder in which the duplicated file will be stored when it is created on the target system.

***TIP:** Duplicating an installed file rather than installing it from multiple points provides better file management because changes to the duplicated file must be made in only one place. You can create multiple duplicates of the same file that is being added, but you must give them each unique names within their destination folders.*

You can perform the following actions on this tab:

- To add a new file to the list of files to be copied during installation, click the *New* icon. If you highlighted a package or a feature in the tree pane, you will be prompted to select a specific component within that package or feature before continuing. Enter the required information on the *Duplicate File* dialog and click *OK*.
- To modify an existing file on the list of files to be copied during installation, highlight the file and click the *Edit* icon. Make the desired changes on the *Duplicate File* dialog and click *OK*.
- To delete a file from the list of files to be copied during installation, highlight the file and click the *Delete* icon.

CREATE FOLDERS

The ***Create Folder*** tab presents a list of folders that will be created on the target system during installation of a Windows Installer package, feature, or component. The following information displays for each folder in the list:

DIRECTORY

This is the full path to the new folder that will be created on the target system during installation

You can perform the following actions on this tab:

- To add a new folder to the list of folders to be created during installation, click the *New* icon. If you highlighted a package or a feature in the tree pane, you will be prompted to select a specific component within that package or feature before continuing. Select the directory on the *Directory* dialog and set permissions if desired. Click *OK*.
- To set or modify permissions for a folder to be created during installation, highlight the folder and click the *Edit* icon. Click the *Permissions* button on the *Directory* dialog, make the desired changes, and click *OK*.
- To delete a folder from the list of folders to be created during installation, highlight the folder and click the *Delete* icon.

ODBC INFORMATION

When you are editing Windows Installer packages, you can view, add, or modify certain ODBC information. The database for MSI packages had five tables relating to ODBC:

- ODBCDataSource
- ODBCSourceAttribute
- ODBCDriver
- ODBCAttribute
- ODBCTranslator

DATA SOURCE SUB-TAB

The *Data Source* tab presents a list of ODBC data sources that will be installed on the target system during installation of a Windows Installer package, feature, or component. (This tab lets you view or modify information relating to the ODBCDataSource and ODBCSourceAttribute tables.) The following information displays for each data source in the list:

DATA SOURCE

This is the internal token name for the data source. It is a primary key for the ODBCDataSource table.

DRIVER

This is the driver associated with the data source specified above. A driver is a program that contains a special interface that allows programs that use the driver to access files in a number of different databases.

***NOTE:** The Windows Installer does not actually perform the ODBC installation, but instead provides the necessary information to the ODBC Driver Manager, which then performs the actual installation.*

You can perform the following actions on this tab:

- To add a new ODBC data source to the list of ODBC data sources to be installed, click the *New* icon. If you highlighted a package or a feature in the tree pane, you will be prompted to select a specific component within that package or feature before continuing. Enter the required information on the *Data Source* dialog and click *OK*.
- To modify an existing ODBC data source, highlight it and click the *Edit* icon. Make the desired changes on the *Data Source* dialog and click *OK*.
- To delete an existing ODBC data source from the list of ODBC data sources to be installed, highlight the data source and click the *Delete* icon.

DRIVER SUB-TAB

The *Driver* tab presents a list of ODBC drivers that will be installed on the target system during installation of a Windows Installer package, feature, or component. (This tab lets you view or modify information relating to the ODBCDriver and ODBCAttribute tables.) The following information displays for each driver in the list:

DRIVER

This is the internal token name for the driver associated with the data source.

FILE

This is the DLL file that contains the ODBC driver software program for the driver specified above.

SETUP FILE

This is the setup DLL file for the driver if it is different from the driver specified above.

***NOTE:** The Windows Installer does not actually perform the ODBC installation, but instead provides the necessary information to the ODBC Driver Manager, which then performs the actual installation.*

You can perform the following actions on this tab:

- To add a new ODBC driver to the list of ODBC drivers to be installed, click the *New* icon. If you highlighted a package or a feature in the tree pane, you will be prompted to select a specific component within that package or feature before continuing. Enter the required information on the *Driver* dialog and click *OK*.
- To modify an existing ODBC driver, highlight it and click the *Edit* icon. Make the desired changes on the *Driver* dialog and click *OK*.
- To delete an existing ODBC driver from the list of ODBC drivers to be installed, highlight it and click the *Delete* icon.

TRANSLATOR SUB-TAB

The *Translator* tab presents a list of ODBC translators that will be installed on the target system during installation of a Windows Installer package, feature, or component. (This tab lets you view or modify information relating to the ODBCTranslator table.) The following information displays for each driver in the list:

TRANSLATOR

This is the internal token name for the translation DLL.

FILE

This is the DLL file for the translator specified above.

SETUP FILE

This is the setup DLL file for the translator if it is different from the Translator specified above.

***NOTE:** The Windows Installer does not actually perform the ODBC installation, but instead provides the necessary information to the ODBC Driver Manager, which then performs the actual installation.*

You can perform the following actions on this tab:

- To add a new ODBC translator to the list of ODBC translators to be installed, click the *New* icon. If you highlighted a package or a feature in the tree pane, you will be prompted to select a specific component within that package or feature before continuing. Enter the required information on the *Translator* dialog and click *OK*.

- To modify an existing ODBC translator, highlight it and click the *Edit* icon. Make the desired changes on the *Translator* dialog and click *OK*.
- To delete an existing ODBC translator from the list of ODBC translators to be installed, highlight the data source and click the *Delete* icon.

SEARCH AND REPLACE FEATURE

WinINSTALL provides a simple facility to search for a character string within a package and replace it with another character string. This feature operates throughout the package, but it is often used within the file operations areas of a package.

For example, perhaps the package specifies that the installers are to add all files to a directory called *OldDir*, but the files now need to go into a directory called *NewDir* instead. The following steps explain how to perform this operation.

1. **Select the desired package in the tree pane and select *Replace* from the *Edit* menu at the top of the console. The *Search and Replace* dialog will appear.**
2. **Under *Sections to Search*, check the checkboxes for sections of the package that you want to search. By default, all of the checkboxes are checked. (In the example given above, you would check only the *File Destination* checkbox).**
3. **In the *Replace All Occurrences …* textbox, type the string that you want WinINSTALL to find. (In the above example, you would enter *OldDir*).**
4. **In the *With This* textbox, type the new string that will be used to replace the string found above. (In the example given above, you would enter *NewDir*).**
5. **Click *OK*. WinINSTALL immediately searches the specified sections of the package and makes the appropriate replacements.**

***NOTE:** This technique does not replace character strings in a text file on the user's workstation. It only replaces strings in the package that will be processed on the user's workstation.*

14

SHORTCUTS

When you highlight a package in the tree pane of the Console and select *Shortcuts* in the list pane, you can use the tabs in the data pane to examine or change the package's instructions regarding shortcuts, also commonly referred to as icons. You can specify the following instructions:

- Shortcuts to be added.
- Shortcuts to be modified, and what modifications to make.

When you highlight a Windows Installer package and select *Shortcuts* in the list pane, there is a single tab in the data pane: *Add*.

ADDING OR CHANGING SHORTCUTS WITH A WINDOWS INSTALLER (MSI) PACKAGE

Windows Installer packages can add new shortcuts and modify existing during a package install. Shortcuts can be configured so that they are automatically installed when the package is processed, or they can be "advertised" so that they are not installed until the shortcut is actually invoked (a.k.a. *installation on demand*). Shortcuts added during an installation are automatically removed during an uninstall.

To manage shortcuts that are added or modified during installation, select a Windows Installer package in the tree pane, highlight *Shortcuts* in the list pane, and click the *Add* tab in the data pane.

ADD TAB

The *Add* tab presents a list of shortcuts that will be added to the target system during installation of a Windows Installer package. The following information displays for each shortcut in the list:

Name • Display name for the shortcut to be added.

Target • The target can be a feature or a file - either the feature that is the current component's parent or one of the files that will be installed with this component.

You can create, delete, edit or view the shortcuts in the list.

To add a new shortcut to the list, click the *Add* icon and enter the required information on the *Shortcut* dialog. It has two tabs - *General* and *Icon*.

To remove a shortcut from the list, highlight it and click the *Delete* icon.

***WARNING:** When you click the* Delete *icon, you will not be asked to verify the deletion. The shortcut will simply be deleted.*

To edit a shortcut in the list, either double-click it or highlight it and click the *Edit* icon. Make the desired changes on the *Shortcut* dialog.

ADD/EDIT SHORTCUT DIALOG

Adding or editing a shortcut will produce the Add or Edit Shortcut dialog, which presents two tabs of information.

GENERAL TAB

The Windows Installer Shortcuts *General* tab allows you to enter the following information:

Name
- The text which appears on the desktop or in a menu where the shortcut is created. It is also the filename portion of the shortcut file itself (i.e., *filename.lnk*).

Shortened Name
- Shortened version of the name (above).

Component
- The component for which the shortcut is being created, selected from a drop-down list. The installation state of this component determines whether the shortcut is created or not.

Create In
- The folder where the shortcut is to be created (can be *@StartMenu*, *@Desktop*, or even an absolute path).

Target (select either Feature or File)
- Feature: the feature that contains the component whose key file is the file launched by the shortcut. When the shortcut is activated, the installer verifies that all the components in the feature are installed before launching this file.
- File: the executable (plus any desired parameters or switches) to launch when the shortcut is executed. Long filenames require quotes.

Description
- The text that will be used for the shortcut tooltip.

ICON TAB

The Windows Installer shortcuts *Icon* tab allows you to enter the following information:

Shortcut Key
- The shortcut key combination to activate this shortcut on the user's desktop (i.e., *Ctrl + x*). You can enter this key combination by actually clicking the desired key combination while the cursor is in the *Shortcut Key* field.

Show Command
- A drop-down offering a selection of window types to open for the launched process (i.e., *Normal*, *Minimized*, *Maximized*).

Arguments
- Any desired command line arguments for the shortcut command line.

Working Directory
- The directory to use as the working directory for the launched process.

Icon
- Click the *Select* Icon button to browse for a file (and an icon within the file) to use as the icon for the shortcut.

15

REGISTRY

WinINSTALL enables packages to specify registry files, keys, or values to be added, removed, or modified. You can also create instructions to set access rights – both user and group rights – for registry values. For instance, you can create a package that installs a registry value that any user can modify or delete using *RegEdit.exe*. Or, you can protect the new registry value being installed by setting a high level of access to it so that only an administrator can delete it from a workstation after it has been installed.

***NOTE:** This security feature is NTFS-specific. Users with FAT drives cannot take advantage of this feature.*

Anyone who modifies registry instructions for a package should be an experienced professional who understands that modifying the registry can have serious consequences.

If you make a mistake that causes problems when starting your computer, you can restore the registry. Follow the instructions given in the *Regedit* Help system for your version of Windows.

REGISTRY FILES

The registry is a hierarchical data store holding many types of information. The registry consists of entries called hives, keys, and values. The hives are the top-level nodes in the hierarchy: *HKEY_CLASSES_ROOT, HKEY_CURRENT_USER, HKEY_LOCAL_MACHINE, HKEY_USERS,* and *HKEY_CURRENT_CONFIG.* All other nodes are keys. Keys can contain none, one, or multiple values or sub-keys. Each value consists of a name and some data in one of several formats.

In addition to the four top-level registry hives (*HKEY_CLASSES_ROOT, HKEY_CURRENT_USER, HKEY_LOCAL_MACHINE, HKEY_USERS, and HKEY_CURRENT_CONFIG*), an additional top-level registry node displays in the WinINSTALL Console: *Follow the per user/per-machine installation*. Values placed under any of the four regular registry hives will be added to those hives in the registry on the target system. Values placed under *Follow the per user/per-machine installation* will be added to either *HKEY_CURRENT_USER* or *HKEY_LOCAL_MACHINE*, based on the installation context specified on the *Install Modes* tab for the package.

REGISTRY CHANGES IN A WINDOWS INSTALLER PACKAGE

Installing Windows Installer packages can add new registry entries to a target system and modify or remove existing ones.

To manage registry entries that are added, modified, or removed during installation, select a Windows Installer package in the tree pane, highlight *Registry* in the list pane, and click either the *Add* or *Remove* tab in the data pane. Each tab has two panes. In the left pane, expand *My Computer* to display the top-level nodes in the registry hive and then navigate to the desired level within the registry as if you were using *Regedit*. The right pane displays registry values - name, data type (such as *DWORD* or *string*), and data – as you highlight registry keys in the left pane.

ADD TAB

On this tab, you create instructions to add registry keys and values during processing of a Windows Installer package.

The *Add* tab presents a hierarchical tree view of the registry entries that will be added to or modified in the registry on the target system when the package is processed. Two panes appear on this tab.

- In the left pane, you can view, add, delete, or rename registry keys to be added to the target system; set permissions for registry keys being added to the target system; or specify new registry values to be added to the target system.
- In the right pane, you can modify the name or data for a registry value being added to the target system; or delete a registry value being added to the target system.

REGISTRY KEYS:

- To specify a new key to be added to a registry hive or key on the target system, right-click the hive or key in the left pane of the tab and select *New Key*. The key is added below the highlighted hive or key, with the temporary name *New Key*. Type the name for the new key (over the temporary name) and press the *Enter* key.
- To delete a registry key from the list of those being added to the target system, right-click the key in the left pane of the tab and select *Delete*.

***WARNING:** When you click* Delete*, you will not be asked to confirm the deletion. The key will simply be deleted.*

- To set access rights for a registry key being added to the target system, right-click the key in the left pane of the tab and select *Permissions*. From the *Permissions* dialog, set the access rights for specific domains, machines, or users.

***TIP:** When permissions have been set for a registry sub-key, a checkmark appears beside* Permissions *on the context menu.*

- To rename a registry key being added to the target system, right-click the key in the left pane of the tab and select *Rename*. The existing name is highlighted and the cursor is blinking. Type the new name and press the *Enter* key.

REGISTRY KEY VALUES:

- To specify a new registry value to be added to the target system, right-click the key in the left pane of the tab and select *New Value*. Enter the value name and select the data type for the new value on the *Add Value* dialog and click *OK*. Enter the string representing the value on the dialog that displays and click *OK*.
- To modify the value name for a registry value being added to the target system, right-click the value name in the right pane of the tab and select *Rename*. Enter the new name for the value in the *Edit Value Name* dialog and click *OK*.
- To modify the value data of a registry value being added to the target system, double-click the value name in the right pane of the tab or right-click the value name and select *Properties*. Enter the new value data on the dialog that displays and click *OK*.
- To delete an existing value for a registry key from the list of values being added to the target system, right-click the value name in the right pane of the tab and select *Delete*.

***WARNING:** When you click* Delete*, you will not be asked to confirm the deletion. The value will simply be deleted.*

REMOVE TAB

On this tab, you create instructions to remove registry keys and values during processing of a Windows Installer package.

The *Remove Phase* tab presents a hierarchical tree view of the registry entries that will be removed from the target system when a Windows Installer package is processed. There are two panes on this tab.

- In the left pane, you can view, add, or delete registry files; view, add, delete or rename registry keys; and add a new value for a registry key.
- In the right pane, you can modify the name or data for an existing registry value or delete an existing registry value.

REGISTRY KEYS:

- To specify a new key to be removed from a registry hive or key on the target system, right-click the key in the left pane of the tab and select *New Key*. The key is added below the highlighted key, with the temporary name *New Key #n*. Type the name for the new key and press the *Enter* key.
- To delete a registry key from the list of those being removed from the target system, right-click the key in the left pane of the tab and select *Delete*.

***WARNING:** When you click* Delete*, you will not be asked to confirm the deletion. The key will simply be deleted.*

- To rename a registry key being removed from the target system, right-click the key in the left pane of the tab and select *Rename*. The existing name is highlighted and the cursor is blinking. Type the new name and press the *Enter* key.

REGISTRY KEY VALUES:

- To specify a new registry value to be removed from the target system, right-click the key in the left pane of the tab and select *New Value*. Enter the value name on the *Add Value* dialog and click *OK*.
- To modify the value name for a registry value being removed from the target system, right-click the value name in the right pane of the tab and select *Rename*. Type the new name over the highlighted existing name.
- To delete an existing value for a registry key from the list of values being removed from the target system, right-click the value name in the right pane of the tab and select *Delete*.

WARNING: *When you click* Delete*, you will not be asked to confirm the deletion. The value will simply be deleted.*

16

System Services

When you highlight a package in the tree pane of the Console and select *Services* in the list pane, you can use the tabs in the data pane to examine or change the package's instructions regarding services.

When you highlight a Windows Installer package and select *Services*, two tabs appear in the data pane:

Add • Lists services that will be added during installation.

Control • Lists services for which you want to give special control instructions, including deletion.

ADDING OR CHANGING SERVICES WITH A WINDOWS INSTALLER (MSI) PACKAGE

WinINSTALL allows the addition of a new service to a target system during the installation of a Windows Installer package and it also enables the controlling (start, stop, delete) of specified system services on the target system during the installation or uninstallation of a package. Installing a system service requires a specific level of security access. The package must be installed under an account with this level of access.

To manage services that are added or controlled when a Windows Installer package is installed, select the package, feature, or component in the tree pane, highlight *Services* in the list pane, and click either the *Add* or *Control* tab in the data pane.

ADDING, EDITING, OR DELETING SYSTEM SERVICES

To install a system service, the package component containing the service must have as its keypath the executable file for the service.

NOTE: *If you have not specified the keypath and try to add a service, you will receive the message "*Component needs to have key path file.*" To proceed, you must highlight a component in the tree pane, select* General *in the list pane, and specify the keypath on the* Summary *tab in the data pane.*

ADD TAB

The *Add* tab presents a list of system services to be added to the target system during installation of the package. The following information displays in the list box on the *Add* tab for each listed service:

SERVICE

This is the name of a system service to be added during installation (and to be removed during an uninstall).

BINARY PATH

The binary path column displays the full path and filename for the executable file used as the keypath for the system service.

SERVICE TYPE

Services must be one of two types:

- Win32 Own Process

A Microsoft Win32 service that runs as its own process.

- Win32 Shared Process

A Win32 service that shares a process.

START TYPE

The Start Type field displays the method by which the service will be started:

- Auto Start

The service starts automatically during system startup. Loads the driver or service when the package is installed.

- Demand Start

The service starts when the service control manager calls the *StartService* function. Loads the driver or service when manually requested by the user.

HOW TO ADD OR EDIT SERVICES

You can add new services to the list or edit or delete existing services in the list.

- To add a service to the list of system services to be added during installation, click the *Add* icon and enter the required information on the *Add Service* dialog. This dialog has two tabs - the *General* tab and the *Options* tab.
- To modify an existing service, highlight it and click the *Edit* icon, or simply double-click the entry in the list. Make the desired changes on *General* and/or *Options* tab of the *Edit*

Service dialog. These tabs are identical to the tabs of the same name on the *Add Service* dialog.

- To remove a service from the list of system services to be added during installation, click the *Delete* icon.

***WARNING:** You will not be asked to confirm the deletion. When you click the* Delete *icon, the service will simply be deleted from the list.*

GENERAL TAB

The *General* Tab provides three fields and three sets of radio buttons.

NAME

The first field is the *Name* field, where you must specify a name for the service being added or edited.

COMPONENT

The *Component* field is a drop-down, allowing you to select the component of the current package with which the service is associated.

RUN SERVICE ARGUMENTS

The *Run Service Arguments* field specifies command line arguments for the service executable.

SERVICE TYPE

The *Service Type* radio buttons indicate that the service will either run as a separate process thread or as a shared process.

START TYPE

The *Start Type* radio buttons specify whether the service will be started automatically when the operating system initializes, or only on demand.

ERROR CONTROL

And the *Error Control* radio button/checkbox combination specifies the handling of startup errors. The three radio buttons represent the following choices:

Ignore	• Log the error and continue with the startup.
Normal	• Log the error, display a message box and continue the startup.
Critical	• Log the error, if possible, and restart with the last known good configuration. If the last known good configuration fails, the startup operation fails.

The state of the *Fail* checkbox indicates whether or not the package installation will fail if the service is unable to start.

OPTIONS TAB

DISPLAY NAME

The Display Name field contains a friendly, more descriptive name to display in the list of services.

LOAD ORDER GROUP

If the service needs to be started as part of a group of services, this field allows placement of this service into a load order group.

DEPENDENCIES

This field allows specification of any dependencies. Such dependencies could be a single system service or a service load order group.

LOGON CREDENTIALS

This tab provides the ability to specify that the service run under either the local system account or a specified logon ID and password. An additional checkbox allows specification of whether or not the service interacts with the desktop of the currently logged on user. Generally, only services run as Local System can interact with the desktop.

IMPORT NT SERVICE

The Import NT Service button allows the selection of a service running on the Console machine. If you select a service running on the local Console machine, its information will be imported into the fields of the *Edit Service* dialog.

CONTROLLING SYSTEM SERVICES

When a Windows Installer package is installed or uninstalled, it may be necessary to start or stop, or even delete a system service during the installation or uninstallation process. Such a service may or may not be one that is to be installed or uninstalled by the package.

WinINSTALL provides the means to specify the required information for this type of control operation from the *Services->Control* tab.

The *Control* tab lists system services to be started, stopped, or deleted during the install or uninstall of the selected package or feature. You can add new services to the list or edit or delete existing services in the list.

- To add a service to the list of system services to be controlled during installation or uninstallation, click the *Add* icon and enter the required information on the *Control Options* dialog.
- To modify an existing service, highlight it and click the *Edit* icon, or simply double-click the entry in the list. Make the desired changes on the *Control Options* dialog.
- To remove a service from the list of system services to be controlled, click the *Delete* icon.

CONTROL OPTIONS

The information that can be specified on the *Control Options* dialog includes the arguments to be passed to the service; whether to start, stop, or delete the service; and whether to wait until the service completes.

Controlling a system service requires a specific level of security access. This means that any package installation or uninstallation that attempts to control a system service must be installed under an account with this level of access. Either the WinINSTALL service account must have administrative privileges or the administrator must grant special elevated privileges for the installation of a package that controls a service.

NOTE: *Service control instructions can apply either to a service you are newly adding or to a service that already exists on the target machine. For this reason, the services listed on the* Control *tab do not have to be the same as those listed on the* Add *tab.*

The *Control* tab presents a list of system services whose behavior on the target system will be changed during installation and/or uninstallation of a Windows Installer package. The following items are available to be specified:

SERVICE NAME

This is the name of the system service for which you are specifying system service commands (*Start/Stop/Delete*).

This name can be found, on a system where the service is installed, under the following key:

`HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Services`

ARGUMENTS

The *Arguments* field enables the addition of command line arguments to the execution of the service startup executable.

WAIT

The *Wait* check box specifies whether or not the installer will wait for the service to complete before proceeding with the installation or uninstallation. If the check box is checked, then the installer will pause until the service successfully starts or stops before continuing with the installation or uninstallation. If *Wait* is unchecked, the installer will pause only until the service state is pending.

SERVICE ACTIONS

For both installation and uninstallation, check boxes allow specification of any or all of three actions to take:

- Start during the StartService action.
- Stop during the StopService action.
- Deleted during the DeleteService action.

17

ADVERTISING

At one time, a product was either installed or not installed. That is, installation was an all-or-nothing process. But today, applications may be installed in a partial fashion, a type of installation often referred to as *advertising*, or *just-in-time installation*.

Advertising means that a product is not actually installed, but instead, the user is provided with certain "advertisements" that the product exists and is available. Perhaps an icon displays on the desktop, or perhaps the product appears on the *Start* menu. When the client clicks the icon or attempts to run the product from the menu, the application is installed at that time.

WinINSTALL allows Windows Installer packages to be configured so that when the package is processed, the product is advertised to the user and installed only when the user first attempts to use it.

Advertising is handled through the *Allow Advertisement* option.

HOW ADVERTISING WORKS

Just-in-time installation is invoked by a client requesting use of an advertised application, through an advertising *entry point*. The client of an application can be either a physical user or another application. An advertising entry point is a method by which a client can request use of a given application. This request is accomplished by invoking one of the application's entry points – for example, by using a shortcut, calling the install through the *Add/Remove Programs* applet (ARP), or programmatically via a COM class.

The way Microsoft Outlook and Microsoft Word (two very discrete applications) interact is a good example of this phenomenon. A physical user who simply double-clicks the Word shortcut (one entry point) can be a direct client of Word . Alternatively, Outlook (an application acting as a client) can use Word by providing the user with the ability to use Word as the email editor. Even though the user may think that he or she is using Outlook to compose the email, Outlook is directly acting as a client of Word by requesting Word's facilities. In this case, the physical user is indirectly using Word by way of a programmatic link between Outlook and Word. This programmatic link is another entry point into Word.

In the example cited above, two entry points are exposed to the actual user:

- Shortcuts
- The entry an application puts into ARP

A WinINSTALL administrator can control how these entry points behave by using the *Shortcuts* section and the *Summary* tab of the package within the Console.

In addition, the programmatic entry points (transparent to the human user) are combined in the *Advertising* section of a package. Typically, there is no reason for a WinINSTALL administrator to edit the programmatic entry points. An exception would be if the administrator is creating a package for an in-house application, developed by his or her own company.

***WARNING:** A WinINSTALL administrator should not need to edit the* Advertising *section of a package. The settings in the* Advertising *section of the package involve programmatic entry points, which are ultimately under control of the programmers who originally created the application. Programmers who require more detailed understanding of advertising Windows Installer packages should consult the Windows Installer documentation (specifically, see the file* msi.chm.*)*

ADVERTISING PACKAGES FOR DISTRIBUTION

Packages that are advertised are not installed until they are used. This avoids saturating the network by distributing software on demand rather than to many machines at once. In other words, advertising can be an effective method of staggering installations to save bandwidth in large enterprises with long recipient lists for software distribution.

In addition, advertising can be a very efficient method of distributing packages. For example, if a particular user has no need for the software, he will never execute it. In other words, such a package will be installed only for those users who actually make use of it.

HOW TO ADVERTISE A WINDOWS INSTALLER PACKAGE

1. **Expand the *Software Distribution* node in the tree pane and select an MSI package. Expand the package and select the desired feature within it.**
2. **Select *General* in the list pane and click the *Advanced* tab in the data pane.**
3. **Under *Advertising* in the middle of the *Advanced* tab, click the checkbox for *Favor Advertising*.**
4. **Click the *Save* icon from the toolbar or choose *Save* from the *File* menu to save the changes you made to the package.**

5 **When this package is chosen for installation, a shortcut, rather than the application itself, will be installed. The application will not actually be installed until the first time the user tries to use the shortcut.**

MANAGING PROGRAMMATIC ENTRY POINTS IN A WINDOWS INSTALLER (MSI) PACKAGE OR MERGE MODULE

Advertising an MSI package or feature takes advantage of the just-in-time, or installation-on-demand functionality that is built in to the Windows Installer. Within the MSI file itself, installation-on-demand is accomplished by means of entry points (i.e., shortcuts, file extensions, and Typelibs) pointing to a parent feature. When an entry point is invoked, all the key paths in that feature are checked prior to launching the application. To the user it appears as if that functionality is already installed. In reality, the Windows Installer does not actually perform the installation until the user tries to start an advertised application or chooses a feature of an application from a toolbar or menu for the first time.

Windows Installer packages offer two types of advertising:

ASSIGNING

Assigning makes changes to the user interface to advertise that the product exists and is available. When the client clicks the icon or asks to run the product from the menu, the application is installed at that time.

PUBLISHING

Publishing does not change the user interface. But it does provide access to other applications which might need to open or use the published application.

When you highlight a Windows Installer package, feature, component, or merge module in the tree pane and select *Advertising* in the list pane, you can add or modify information related to programmatic entry points to a package, feature, or merge module.

***NOTE:** If a package, feature or merge module is highlighted in the tree pane and* Advertising *is selected in the list pane, you must select a specific component within that package, feature or merge module before adding, modifying or deleting information on tabs in the data pane.*

Much of the information appearing in the data pane is used by the Windows Installer to locate the component when it is actually installed. Some tabs contain information required for accessing COM objects, and some are specific to the .NET platform. Information in the *HKEY_CLASSES_ROOT* hive of the registry maps the identifiers of COM entities (CLSIDs) to the physical locations of the modules that contain them (file locations).

Follow these steps to enter or modify information related to programmatic entry points to a Windows Installer package, feature or merge module:

1. **Navigate to the Windows Installer package, feature, component, or merge module under *Software Distribution* in the tree pane.**
2. **Select *Advertising* in the list pane.**
3. **In the data pane, select one of the following tabs:**

- **The *TypeLib* tab lets you view, add, or modify information related to the TypeLib property of a component. (TypeLIb table)**
- **The *Classes* tab lets you view, add, or modify information related to the class identifier (CLSID) of a COM class. (Classes table)**
- **The *ProgID* tab lets you view, add, or modify information related to the programmatic identifier (ProgID) of a COM class. (ProgID table)**
- **The *Extensions* tab lets you view, add, or modify information related to the Shell file extension associations of a resource component. (Extensions table)**
- **The *AppID* tab lets you view, add, or modify information related to the Application Identifiers for COM servers. (AppID table)**
- **The *Assembly* tab lets you view, add, or modify information related to the assemblies in which component code must reside in order for .NET to execute the code at runtime.**
- **The *AssemblyName* tab lets you view, add, or modify information related to the name of the .NET assembly that holds the code of a component.**

TYPELIB TAB

Every CLSID (component) in the *HKEY_CLASSES_ROOT\CLSID* hive of the registry has a *TypeLib* subkey. Type information provides information about supported interfaces and methods, and allows clients to invoke these methods dynamically. A type library describes all of the interfaces supported by the (server) object, all of the methods, and the parameters of those methods. Since a type library describes exactly what an interface can do, it can be used to marshal data between processes. An associated feature must be installed in order for the type library to be operational. The information on this tab is used to populate the TypeLib table in the MSI database.

The *Typelib* tab presents a list of entries. Clicking the *Add* icon or selecting an existing entry and clicking the *Edit* icon, displays the *Typelib* dialog, which offers the following fields:

LibID • The Globally Unique Identifier (GUID) that identifies the type library.

Component • The GUID for the component associated with the selected TypeLib.

Description • Textual description of the type library.

Help path • Drop-down selection of the folder where help for the type library will be installed.

Language • This drop-down allows the selection of the language of the type library.

Dialect • Drop-down providing a selection of dialects, dependent on the selected language.

Version • Version of the type library, including both major and minor versions.

Cost • Amount of disk space in bytes, associated with the registration of the type library. Must be a non-negative number or null.

CLASSES TAB

Class information about a component is needed to support COM functionality. Every COM class has a unique Class ID (CLSID), which is a 128-bit integer that uniquely identifies the component. The CLSID is used whenever an instance of that component is created. Class information is used to populate the Class table in the MSI database.

The *Classes* tab presents a list of entries. Clicking the *Add* icon or selecting an existing entry and clicking the *Edit* icon, displays the *Class* dialog, which displays the class data fields on two tabs, *General* and *Options*.

The following fields display on the *General* tab:

Component • Drop-down offering the 128-bit Globally Unique Identifiers (GUIDs) that identify the components of the currently selected feature. The selected GUID identifies the component whose key file provides the COM server.

Feature • Drop-down offering the descriptions of each feature in the current package.

CLSID • GUID identifying the CLSID key in the registry.

Context • The server context for this COM server (may be one of the following values: *LocalServer*, *LocalServer32*, *InprocServer*, *InprocServer32*).

Default Inproc Handler • Available as an option only if the context is set to *LocalServer* or *LocalServer32*. 1 indicates a 16-bit InprocHandler, 2 indicates a 32-bit InprocHandler, and 3 indicates both 16 and 32 bit InprocHandlers.

Default Program ID • Every insertable object class has a programmatic identifier (ProgID). The value you enter here will be used as the default if no other ProgID is supplied. ProgIDs have the format: *<vendor>.<component>.<version>*.

Clicking on the *Options* tab will display the following data fields:

File Type Mask • The file mask for the HKEY_CLASSES_ROOT entries in the registry. Multiple patterns must be delimited by semicolons.

Arguments • Available as an option only when Context LocalServer or LocalServer32. The text in this field is registered as the argument against the OLE server and is used by OLE for invoking the server.

Description • This is a human readable name for the component class represented by the CLSID. This is the name that will be displayed in the user interface.

Icon • The icon associated with this CLSID. By default, the icon comes from the COM server. Advertised file associations and shortcuts require a specified icon to display properly.

PROGID TAB

The program identifier (ProgID) is a textual version of a COM class's name - easier to understand than the CLSID. ProgIDs are not necessarily unique. The recommended format for a ProgID is *SERVER.COMPONENT.VERSION*. Entering information about the program identifier is the first part of creating the registration information for defining a file type. Program identifier information entered is used to populate the ProgID table, which provides the Windows Installer with necessary COM information.

The *ProgID* tab presents a list of entries. Clicking the *Add* icon or selecting an existing entry and clicking the *Edit* icon, displays the *ProgID* dialog, which offers the following fields

ProgID • The program identifier (Program ID).

ProgID Parent • Defined only for version independent program IDs, this is the program ID for the parent COM class.

Class • Optional, but only available for version independent Program IDs. This is a CLSID from within the same package.

Description • Optional description of the Program ID.

Icon • Optional icon for the Program ID.

EXTENSIONS TAB

Extensions are used to create a file association for an application.

The *Extensions* tab presents a list of entries. Clicking the *Add* icon or selecting an existing entry and clicking the *Edit* icon, displays the *Extension* dialog, which displays the extensions data fields on three tabs, *Extension, Mime,* and *Verb*.

The following fields display on the *Extension* tab:

Extension • The file name extension being registered (i.e., the extension for which the component specified below will act as a file name extension server). The file name extension does not include the preceding period and can be up to 255 characters long.

Component • Component associated with the extension.

Feature • Drop-down to select the description of the feature associated with the extension.

Associated Program ID • Drop-down to select the program ID (text description) associated with the file name extension specified above.

MIME Content Type • The MIME content type (expressed in the form of type/format) to be written for the extension. An example of a MIME content type for a normal ASCII file is text/plain.

The following fields are available on the *MIME* tab:

MIME • The particular MIME (Multipurpose Internet Mail Extension) type associated with this extension.

Associated Class • CLSID of the COM server to be associated with the specified MIME type.

The following fields display on the *Verb* tab:

Verb • Action to invoke (i.e., *open*, *print*, etc.).

Command • Text to display on the context menu to indicate this action.

Arguments • Command arguments. For example, %1 to pass the filename.

APPID TAB

Application Identifiers (AppIDs) are unique identifiers for COM servers. Information stored in the AppID table in the MSI database is used by the installer to configure and register DCOM servers to take one of the following actions:

- Run the DCOM server as a service.
- Run the DCOM server under a different identity than the user who activated the server.
- Register the DCOM server so that it is activated on a different computer.
- Configure the default security access for the DCOM server.

The *AppID* tab presents a list of entries. Clicking the *Add* icon or selecting an existing entry and clicking the *Edit* icon, displays the *AppID* dialog, which offers the following fields:

AppID
- 128-bit Globally Unique Identifier (GUID) identifying the *AppID* key in the registry.

Remote Server Name
- The name of a server that will be installed on client machines to configure the client to request the object be run at a particular machine whenever an activation function is called for which a COSERVERINFO structure is not specified. This allows client applications to be written without hard-coded server names and then configured by modifying this value for the classes of objects they use.

Local Service
- Value specifying to configure and register the DCOM server to run as a local service with this name.

Service Parameters
- Parameters to be passed to the service specified above.

DLL Surrogate
- Allows DLL servers to run in a surrogate process. If no value is entered, the system-supplied surrogate is used; if a value is entered, it specifies the path of the surrogate to be used.

Activate At Storage
- Checkbox to indicate whether or not to set the COM server's ActivateAtStorage registry value to "Y."

Run As Interactive User
- Checkbox to indicate whether or not to run this server as the currently logged on interactive user.

ASSEMBLY TAB

An assembly is a logical collection of one or more EXE or DLL files containing an application's code and resources. An assembly also contains a manifest, which is a metadata description of the code or resources inside the assembly. All security, namespace resolution and versioning features work on a per-assembly basis. An assembly often resides in a single file, but it can also be a logical - not physical - collection of multiple files in the same directory. Assemblies are used by the .NET framework to maintain a set of related resources for a feature or product so that they cannot be "broken" by the installation or uninstall of another feature or product.

The *Assembly* tab presents a list of entries. Clicking the *Add* icon or selecting an existing entry and clicking the *Edit* icon, displays the *Assembly* dialog, which offers the following fields:

Component
- Drop-down to select the name of the associated component.

Feature
- Drop-down to select the name of the associated feature.

File Manifest • Contained within an assembly, this file is a metadata description of the code or resources inside the assembly.

File Application • The EXE or DLL containing the resource code for the component or feature associated with this assembly.

Assembly Type Radio Buttons • Specify whether the assembly is a .NET Framework or Win32 assembly.

ASSEMBLY NAME TAB

An assembly is a logical collection of one or more EXE or DLL files that contain an application's code and resources (see above). An assembly's name has a significant impact on the assembly's scope and use by multiple applications. Assemblies intended for use by one application only require a name that is unique within the application. Assemblies intended to be shared by multiple applications must be signed with strong names using standard public key cryptography. Signing an assembly with a strong name allows the assembly to be deployed in the global assembly cache.

The *Assembly Name* tab presents a list of entries. Clicking the *Add* icon or selecting an existing entry and clicking the *Edit* icon, displays the *Assembly Name* dialog, which offers the following fields:

Component • The component whose key file is a resource in the assembly.

Name • The name of the attribute associated with the value specified below. The file name of an assembly in the global assembly cache must match the assembly name (not including the file name extension, such as .exe or .dll). Private assemblies deployed only in the root application directory can have an assembly name that is different from the file name.

Value • The value associated with the name specified above.

18

File Edits

WinINSTALL provides the means to edit INI files and ASCII and Unicode text files as part of package installs. ASCII and Unicode text file edits are not natively supported by the Windows Installer, but WinINSTALL adds this capability to any Windows Installer package it creates.

When you select a package, feature, or component in the tree pane and then select *Edits* in the list pane, the data pane will display two tabs: *INI Files* and *ASCII Files*.

The *INI Files* and *ASCII Files* tabs both provide *Add* and *Remove* sub-tabs. The *Add* sub-tab provides instructions to the installer for adding files and text during package installation, while the *Remove* sub-tab provides instructions on removing text during package installation.

INI File Edits

Windows INI files are text files containing one or more headings enclosed in square brackets. Each heading is followed by one or more lines in the format *key=value*. Each key beneath a given heading must be unique. (In legacy *SYSTEM.INI* files, the *Device=* lines beneath the *[386Enh]* section are an exception to this rule). Neither the order of the headings nor the order of the *key=value* statements beneath any heading is significant; only their presence or absence matters.

If the file you want to edit does not follow these rules, you cannot edit it through the *INI Files* tab. In this case, you can remove it from the *INI Files* tab, move it to the *ASCII Files* tab, and edit it there.

WinINSTALL lets you make the following types of edits to INI files:

- Add sections or keys to .ini files
- Delete sections or keys from .ini files
- Remove entire lines from .ini files
- Search and replace strings in .ini files

During the install process, the installer checks the INI file for each section specified to be modified. It will take the following actions:

- If the heading does not exist, the installer will create it and add the specified lines beneath it.

- If the heading does exist, the installer will check for the specified keys beneath the heading. If a specified key exists, the value will be changed to the one specified in the *package*. If the key does not exist, the *key=value* statement will be added.

TEXT FILE EDITS

WinINSTALL enables the creation and editing of ASCII and Unicode text files. Supported operations include the insertion of text at specified locations within such files. For example, the text can be inserted before or after other, specified, text, or at the beginning or end of the file. Text file editing capabilities also include removing text and removing entire lines in text files.

WinINSTALL also provides the capability, when editing the legacy file *Autoexec.bat*, to add directories to the *PATH=* line.

EDITING FILES WITH A WINDOWS INSTALLER (MSI) PACKAGE

A Windows Installer package can add and modify ASCII and Unicode text files, INI files, and environment variables on a target machine.

To manage edits to text or INI files or environment variables during installation, select a Windows Installer package in the tree pane, highlight *Edits* in the list pane, and click the *INI Files*, *ASCII Files*, or *Environment* tab in the data pane.

INI FILES TAB

When you click the *INI Files* tab in the data pane, both the *Add* and *Remove* sub-tab are split into two panes, with the tree view of INI files on the left and the detail for changes to be made on the right. In the left pane, there is one top-level node - I*NI File Values*. The following information displays in the right pane:

- Key
- Value
- Action

ADD AND REMOVE SUB-TABS

The *Add* sub-tab lets you examine details or provide instructions about how a Windows Installer package will add or modify INI files on a target machine during installation.

The *Add* sub-tab specifies how a Windows Installer package will add or modify INI files on a target during installation. The *Remove* sub-tab provides instructions on headings, keys and values to remove from INI files during an installation. These two tabs are organized identically, and all options are provided on both tabs--but the entries on the *Add* tab will be added, while the entries on the *Remove* tab will be removed during an install of the package. During an uninstall, the entries on the *Add* tab will be removed, and the entries on the *Remove* tab will be ignored.

***TIP:** Entries on the* Add *tab will be added during an install of the package and removed during an uninstall of the package. In contrast, the entries on the* Remove *tab will be removed during an install operation, but they will be ignored during an uninstall of a Windows Installer package.*

You can perform the following actions on these tabs:

- Specify an INI file to be created or modified on the target system.
- Delete an INI file from the list of files to be created or modified.
- Specify a section to be created or modified in an INI file on the target system
- Specify a key to be added, removed, or modified in a section of an INI file on the target system
- Edit an existing INI file on the target system. Edits can perform one of three actions:
  - Add a line (creates or updates an entry).
  - Create a line (creates an entry only if it does not already exist).
  - Add a tag to an existing key (creates a new entry or appends a new comma-separated value to an existing entry).

HOW TO SPECIFY AN INI FILE TO BE ADDED TO OR MODIFIED ON THE TARGET SYSTEM

You would add a new INI file for one of two reasons - you want the package to add an INI file that doesn't currently exist on the target system or you want the package to make changes to an INI file that already exists on the target system. To add a new file:

1. **Right-click *INI File Value* node on the *Add* or *Remove* tab.**
2. **Select *New File* from the pop-up menu.**
3. **On the *INI File Name* dialog, enter or browse for the name of the file to add and click *OK*.**

HOW TO DELETE A FILE FROM THE LIST OF INI FILES TO BE CREATED OR MODIFIED

1. **Right-click the file and select *Delete* from the context menu.**
2. **The selected file will no longer appear beneath the *INI Files* node.**

HOW TO SPECIFY A SECTION TO BE ADDED TO OR MODIFIED

You would add a new section to an INI file for one of two reasons - you want the package to add a section that doesn't currently exist on the target system or you want the package to make changes to a section that already exists on the target system. To add a new section:

1. **Right-click the INI file to which you want to add the section.**
2. **Select *New Section* from the pop-up menu.**
3. **In the box that appears next to the new node under the INI file, enter the section name.**

HOW TO SPECIFY A KEY TO BE ADDED TO OR MODIFIED

You would add a new key to an INI file for one of two reasons - you want the package to add a key that doesn't currently exist on the target system or you want the package to make changes to a key that already exists on the target system. To add a new key:

1. **Right-click the desired INI file section on the left side of the *Add* or *Remove* tab.**
2. **Select *New Key* from the pop-up menu.**
3. **Add the desired key and value to the *Properties* dialog, select the appropriate radio button for the desired operation (*Add Line*, *Create Line*, or *Add Tag*), and click *OK*.**
4. **The new key will appear in the right pane of the tab. You can add multiple keys to a section.**

HOW TO EDIT A KEY TO BE ADDED

1. **In the left pane, expand the desired INI file and section.**
2. **In the right pane, right-click the desired key.**
3. **Select *Rename* from the pop-up menu.**
4. **Complete the *Properties* dialog (change the name of the key, its value, and/or the action that will be taken--*Add Line*, *Create Line*, *Add Tag*) and click *OK*.**
5. **The modified information for the key will displays in the right pane.**

ASCII FILES TAB

The *ASCII Files* tab enables you to specify edits to ASCII and Unicode text files. When you click the *ASCII Files* tab in the data pane, you can click either the *Add* or the *Remove* sub-tab. Both sub-tabs are split into two panes, with the tree view of ASCII files on the left and

the detail for changes to be made on the right. In the left pane, the top-level node - *ASCII Files* – displays.

ADD AND REMOVE SUB-TABS

The *Add* sub-tab specifies how a Windows Installer package will add or modify ASCII or Unicode text files on a target machine during installation. The *Remove* sub-tab provides instructions on removing text from files during an installation; these instructions are ignored on uninstallation.

The *Add* and *Remove* tabs are organized identically, and all options are provided on both tabs--but the entries on the *Add* tab will be added during an install of the package, while the entries on the *Remove* tab will be removed during an install or uninstall of the package.

***TIP:** Entries on the* Add *tab will be added during an install of the package and removed during an uninstall of the package. In contrast, the entries on the* Remove *tab will be removed, but only when the operation is an install. These entries are ignored during an uninstall of a Windows Installer package.*

You can perform the following actions on these tabs:

- Add a new ASCII or Unicode text file to the list of files to be created or modified on the target system.
- Delete a file from the list of text files to be created or modified on the target system.
- Edit a text file in the list of text files to be created or modified on the target system:
  - Insert text.
  - Find and replace text strings.
  - Modify the path in the *Path*= line in a legacy *AUTOEXEC.BAT* file.
- Change or delete a listed text file edit.

HOW TO ADD A NEW FILE TO THE LIST OF TEXT FILES TO BE CREATED OR MODIFIED

You would add a new ASCII or Unicode file for one of two reasons - you want the package to add an ASCII file that doesn't currently exist on the target system or you want the package to make changes to an ASCII file that already exists on the target system. To add a new file, follow these steps:

1. **Right-click the top-level node - *ASCII Files* - in the left pane and select *New File* from the context menu.**

2 **On the T*ext File Name* dialog, browse to or enter the fully-qualified UNC path to the filename of the file you are adding and click *OK*.**

If you selected a package or a feature in the tree pane, you will be prompted to select a specific component within that package or feature before continuing.

3 **The *ASCII Files* node will expand to display the new file beneath it.**

HOW TO DELETE A FILE FROM THE LIST OF TEXT FILES TO BE CREATED OR MODIFIED

1 **Right-click the file and select *Delete* from the context menu.**

2 **The selected file will no longer appear beneath the *ASCII Files* node.**

HOW TO EDIT A TEXT FILE ON THE TARGET SYSTEM

You can perform three kinds of edits on text files:

- Insert text

  1 **Add the target file to the list if it is not already listed (see above for steps to add a file to the list).**

  2 **Right-click the file and select *New Edits* from the context menu.**

  3 **On the *Text File Edits* dialog, enter the new text to insert, optionally enter *Position Text* (text after or before which the new text is to be added) select the appropriate *Action* radio button to specify where the new text should be inserted, and click *OK*.**

- Find and replace text strings

  1 **Add the target file to the list if it is not already listed (see above for steps to add a file to the list).**

  2 **Right-click the file and select *New Edits* from the context menu.**

3. **On the *Text File Edits* dialog, enter the new text to add, enter the old text to be replaced, and select the *Action* radio button marked *Replace* to specify a search and replace operation.**
4. **Click *OK*.**

- Add a path to the *Path*= line in a legacy *AUTOEXEC.BAT* file.

1. **Add the target file to the list if it is not already listed (see above for steps to add a file to the list).**
2. **Right-click the file and select *New Edits* from the context menu.**
3. **On the *Text File Edits* dialog, enter the new path to be added (or multiple paths, separated by semicolons), optionally enter *Position Text* (an existing path after which to add the new path), select the *Action* radio button marked *Path Addition*, and click *OK*.**

HOW TO CHANGE OR DELETE A LISTED TEXT FILE EDIT

1. **Expand the top-level node - *ASCII Files* - in the left pane and select the desired file.**
2. **Right-click the desired edit in the list in the right pane and select the desired operation:**
   - **Select *Rename* to display the *Text File Edits* dialog, where you can modify the edit as desired.**
   - **Select *Delete* to remove the edit from the list in the right pane.**

ENVIRONMENT TAB

On the *Environment* tab, you can provide information that will be used to modify environment variables on the target system during installation of a Windows Installer package, feature or component. An environment variable defines some changeable aspect of a user's working environment, such as the default printer, browser, or text editor to be used. Environment variables are generally set during login.

CREATING AND MODIFYING ENVIRONMENT VARIABLES

To add or modify environment variable information, you click the *Add* or *Edit* icon on the list to display the *Environment Variable* dialog. The following information is available on this dialog:

Variable • The environment variable to be created or modified.

Value • The value to which the environment variable is to be set.

Function • The actual operation: *Set*, *Create*, or *Remove*.

Attributes • Checkboxes--select any or all: *System*, *Remove on uninstall*, *Append value to original*, *Prepend value to original*.

Separator character • Default separator is a semicolon. This character will be used for append or prepend operations.

HOW TO ADD AN ENVIRONMENT VARIABLE TO THE LIST

To add an environment variable to the list of environment variables to be created or modified during package installation, click the *New* icon. Enter the necessary information on the *Environment Variable* dialog (see above) and click *OK*.

HOW TO MODIFY A LISTED ENVIRONMENT VARIABLE

To modify an existing environment variable, select the desired variable in the list and click the *Edit* icon. Change the desired information on the *Environment Variable* dialog (see above) and click *OK*.

HOW TO DELETE A LISTED ENVIRONMENT VARIABLE

To remove an environment variable from the list of environment variables to be created or modified during package installation, select the desired variable in the list and click the *Delete* icon..

19

MERGE MODULES

A merge module (.MSM file), like a Windows Installer package (.MSI file), is a database containing instructions, components, and setup logic. A merge module, however, cannot stand alone. Instead, it must be merged into an existing Windows Installer package. This process alters the original package by adding the merge module's components and logic to it.

WinINSTALL enables the use of the *Discover Wizard* to create a merge module (for details, see Using Discover to Build a Windows Installer Package or Merge Module in the Building Packages chapter of this guide). Once created, a merge module can be edited in the WinINSTALL Console in the same way as a Windows Installer package. Merge modules are located in designated folders, accessible under a separate tree node, *Merge Module Folders* under *Software Distribution* in the tree pane. The tabs in the data pane are very similar to, but not entirely identical with, those for MSI packages.

MERGE MODULE CACHE

WinINSTALL enables the creation of a merge module "cache." The cache acts as a quick lookup table for the MSI file creation process for both the *Discover Wizard* and the console. The cache contains critical information about the merge modules it contains. When the WinINSTALL converter is generating an MSI file from a WinINSTALL package, or Discover is creating a new MSI file, it can look at each file in a package and see if the file is part of a merge module in the merge module cache. If a file is distributed as part of a Microsoft (or other) merge module, the converter can include that module instead of the individual file. This approach prevents conflicts with other applications that distribute the same file and allows applications to be more compliant with Microsoft (MSI) applications that use merge modules. This process effectively emulates a developer making a conscious decision to merge a redistributable module into a Windows Installer package without the administrator having to think about it.

***NOTE:** When a merge module is added to the "cache," the files within that merge module are listed in the file* MergeModule.xml *on the WinINSTALL share. When an MSI package is built or converted, WinINSTALL compares this file list with the files in the package. When it finds a match, it uses the merge module, rather than including the individual files in the package. This approach prevents the creation of multiple entries with different GUIDs for the same common components.*

WinINSTALL installs a *Microsoft Merge Modules* sub-folder containing a large number of redistributable Microsoft merge modules. You can add additional and updated .msm files to this folder, and you can create other folders beneath the *\MergeModules* folder.

The most common activities associated with merge modules are to add additional vendor-supplied .msm files to the merge module cache and then to include them in new Windows Installer packages.

ADDING NEW MERGE MODULE FOLDERS TO THE CACHE

To add new merge module folders to the cache, first create the desired folder beneath the *\MergeModules* folder in the share and copy the desired .msm files into that folder. Then in the console, right-click the *Merge Module Folders* node beneath *Software Distribution* in the tree pane and select *Add Folder*.

Browse to the new folder, add a descriptive name for the folder and click *OK*. WinINSTALL will add the new folder and its contents to the Console.

UPDATING THE CACHE

To add new merge modules to an existing merge module folder, copy the desired .msm files to the appropriate MergeModule sub-directory in the WinINSTALL share.

Right-click this folder beneath the *Merge Module Folders* node in the tree pane and select *Update Cache*. WinINSTALL will update the XML list to include the changes to the cache folder contents.

EDITING MERGE MODULES

Editing a Windows Installer merge module is very similar to editing a Windows Installer package.

With the exception of the *General* category, the tabs in the data pane for the other categories are the same for both merge modules and packages. Therefore, only the tabs in the data pane

for the *General* category of merge modules are explained in this chapter. For an explanation of the other categories, please refer to the following chapters:

Files

Shortcuts

Registry

System Services

File Edits

Advertising

When you select a merge module in the tree pane and choose *General* in the list pane, four tabs appear in the data pane:

- Summary Tab
- Advanced Tab
- Merge Module Tab
- Directory Tab

SUMMARY TAB

When you highlight a Windows Installer merge module in the tree pane, select *General* in the list pane, and click the *Summary* tab in the data pane, you can view, enter, edit or delete general information about the package, such as product information, support information, and whether and how the application should display in *Add/Remove Programs* in the Control Panel.

On the *Summary* tab, you can view the following information:

MSI

This field displays the version of the MSI database schema to be used to create the merge module. For existing merge modules, the version is supplied automatically based on the MSI version installed on the PC where the merge module was created and cannot be changed on this tab.

MODULE SIGNATURE

The *Module Signature* is the GUID that identifies the merge module.

MANUFACTURER

The *Manufacturer* is the software vendor which created the application.

PACKAGE CODE

The *Package Code* is the GUID that serves as the principal identifier for the package. The package code must be unique for every package. A small update, a minor upgrade, or a major upgrade of the same application all require separate package codes. Versions of the same application in different languages also require separate package codes.

Click the *Ellipsis* button to browse for the package code. If no GUID exists, a message box offers to generate a new GUID.

COMMENTS

The *Comments* field holds descriptive text that will display when a user selects the application on the *Add/Remove Programs* dialog (available from the Control Panel).

ADVANCED TAB

The *Advanced* tab holds installation information, such as the components within the merge module, the conditions that must be met in order for the merge module to be installed, the sequence of actions that will take place during a regular install, an administrative install, and an advertised install, and how the installer will locate files.

The *Advanced* tab consists of six sub-tabs. Because these sub-tabs are the same as those provided for Windows Installer packages, they are all described fully elsewhere in this guide (links provided below):

Components Sub-tab.

Launch Condition Sub-tab.

AppSearch Sub-tab.

Dialogs Sub-tab.

Custom Actions Sub-tab.

Package History Sub-tab.

COMPONENTS SUB-TAB

Windows Installer merge modules, like packages, are made up of components. Components hold various files, shortcuts, and/or registry entries. An explanation of read-only items on the *Components* tab appears in the Components Sub-tab section of the Additional MSI Package Options chapter, earlier in this guide.

LAUNCH CONDITION SUB-TAB

Windows Installer merge modules can include special launch conditions which must be met in order for the merge module to be processed. Full details on the information on the

Launch Conditions sub-tab are presented in the Windows Installer Launch Conditions section of the Package Conditions chapter, earlier in this guide.

APPSEARCH SUB-TAB

The information on the AppSearch tab tells the installer what to look for to determine where an application should be installed. Full details on this information are provided in the AppSearch Sub-tab section of the Additional MSI Package Options chapter, earlier in this guide.

DIALOGS SUB-TAB

The information on the *Dialogs* tab tells the installer what dialogs to display when an application is installed. Full details on this information are provided in the Dialogs Sub-tab section of the Additional MSI Package Options chapter, earlier in this guide.

CUSTOM ACTIONS SUB-TAB

The information on the *Custom Actions* tab tells the installer what custom actions to execute when an application is installed. Full details on this information are provided in the Custom Actions Sub-tab section of the Additional MSI Package Options chapter, earlier in this guide.

PACKAGE HISTORY SUB-TAB

If the *Always log changes* checkbox has been checked, the information on the *Package History* tab will reflect the changes made to the merge module in the WinINSTALL Console, including the username of the user who made each change. Details on this information are provided in the Package History section of the Additional MSI Package Options chapter, earlier in this guide.

MERGE MODULE TAB

When you highlight a Windows Installer merge module in the tree pane, select *General* in the list pane, and click the *Merge Module* tab, three sub-tabs appear in the data pane – *Signature*, ***Sequence***, and *Ignore Tables*.

SIGNATURE SUB-TAB

On the *Signature* sub-tab, you can view or modify the following general information for the merge module:

Module ID • Descriptive name of the merge module itself.

GUID • The *Globally Unique Identifier*, a 128-bit unique identification string, which identifies the current merge module. If you click the *Ellipsis* button, WinINSTALL will offer to create a new GUID.

Language • The language for which this merge module is to be used (* means any language).

Dialect • Drop-down offering choices based on the selected language.

Version • Numeric version number for the current merge module.

Exclusions • A list of other merge modules that are incompatible with this merge module. Click the *Add* icon to add a merge module to this list. To delete a module from the list, select the desired module and click the *Delete* icon.

Dependencies • A list of other merge modules required by this merge module. Click the *Add* icon to add a merge module to this list. To delete a module from the list, select the desired module and click the *Delete* icon.

SEQUENCE SUB-TAB

There are three possible types of MSI installations - a normal installation, an administrative installation, and an advertised installation. Each type of installation has two different sequences of actions - one for the user interface and one for the actual installation of the product. Within each sequence, each separate action has a name, an optional condition, and a sequence number that determines the order of execution. Full details on this information are provided in the Sequence Sub-tab section of the Additional MSI Package Options chapter, earlier in this guide.

IGNORE TABLES SUB-TAB (OF THE MERGE MODULE TAB)

When you highlight a Windows Installer merge module in the tree pane, select *General* in the list pane, click the *Merge Module* tab in the data pane and then click the *Ignore Tables* sub-tab, you can view the list of tables in the merge module that will be ignored (i.e., not merged) when the merge module is merged into an MSI file.

If a table in the merge module is listed on the Ignore Tables sub-tab, it will not be merged into the .msi file. If the table already exists in the .msi file, it will not be modified by the merge.

The tables in the ModuleIgnoreTable can therefore contain data that is unneeded after the merge.

***TIP:** To minimize the size of the merge module, it is recommended that unused tables be removed from merge modules, rather than listing these tables on the* Ignore Tables *Sub-tab.*

You can perform the following operations on the *Ignore Tables* sub-tab:

Add a new table to the list
- click the *Add* icon and at the bottom of the list, where the cursor appears, type in the name of the table to be ignored.

Delete a table from the list
- highlight the desired table name in the list and click the *Delete* icon.

***NOTE:** You will not be asked to confirm the deletion.*

DIRECTORY TAB

When you highlight a Windows Installer merge module in the tree pane, select *General* in the list pane, and click the *Directory* tab in the data pane, you can enter, view or edit all of the directories in the merge module. This information is the same as that available for Windows Installer packages, and it is described in detail in the Directory Tab section of the Additional MSI Package Options chapter, earlier in this guide.

Section 4

Additional Operations

Chapter 20: Reports

Chapter 21: Miscellaneous

20

REPORTS

Reports present useful and meaningful views of information in the WinINSTALL database. Reports are organized into the following nine categories, eight of which come with a wide array of ready-to-use reports for a specific functional area of the product and one (*Custom*) that is intended for reports you create with Crystal Designer.

- Conflict Assessment
- Console
- Custom

The data that displays in the reports is dependent upon the database to which the Console is connected and the actions that have been performed in the console. For example, some Conflict Assessment reports require that Conflict Assessments have been created, while others require that they have also been run.

To manage reports in the console, expand the *Reports* node in the tree pane and select a specific report category beneath it. All reports in the selected report category are listed on the *Reports* tab in the data pane.

The following information displays for each report in the list:

Name • The title of the report

Description • A brief description of the purpose and content of the report

On the *Reports* tab, you can perform the following actions:

- Add a report to the Console.
- Modify the properties of a report.
- Delete a report.
- Run a report.
- Print a report.
- Launch Crystal Designer.

The following sections describe each of these activities in detail.

HOW TO ADD A REPORT TO THE CONSOLE

To add a report to the Console, you must have first created the new report with Crystal Designer or retrieved the report from the Attachmate web site or some other source. If you have a new report to add, follow these steps to cause the new report to appear in the WinINSTALL Console.

1. **Expand the *Reports* node in the tree pane and select a report category beneath it. You can add a new report to any report category.**
2. **Do one of the following:**
   - **Click the *Add* button on the data pane.**
   - **Click *Add Report* icon on the toolbar.**
   - **Select *Add Report* from the *Reports* menu.**
3. **Enter the required information on the *Report Definition* dialog.**
4. **The new report will now display on the data pane, at the bottom of the list of reports in the selected report category.**

HOW TO MODIFY REPORT PROPERTIES

1. **Expand the *Reports* node in the tree pane and select the report category containing the report you want to modify.**
2. **Select the desired report in the data pane.**
3. **Do one of the following:**
   - **Click the *Properties* button on the data pane.**
   - **Click *Edit Report* icon on the toolbar.**
   - **Select *Edit Report...* from the *Reports* menu.**
4. **Modify the information on the *Report Definition* dialog as desired.**

   After modifying a report, if you click the report category containing the modified report, the new title and description will display on the data pane (if you changed this information) and the modified report will display when you click the *Run* button.

HOW TO DELETE A REPORT

You can delete any report that you have added to the Console. You cannot delete the reports which are supplied with WinINSTALL.

1. **Expand the *Reports* node in the tree pane and select the report category that holds the report that you want to remove.**
2. **Select the desired report in the data pane.**
3. **Do one of the following:**
   - **Click the *Delete* button on the data pane.**
   - **Click the *Delete Report* icon on the toolbar.**
   - **Select *Delete Report* from the *Reports* menu**

   The report will no longer be displayed in the list of reports in the data pane. The associated report file will also have been deleted from the Console machine.

***NOTE:** You can remove only those reports that you added to the Console. You cannot remove a report provided by WinINSTALL.*

HOW TO RUN A REPORT

1. **Expand the *Reports* node in the tree pane and select the report category where the report that you want to run is located.**
2. **Select the desired report in the data pane.**
3. **Do one of the following:**
   - **Double-click the report.**
   - **Click the *Run* button on the data pane.**
   - **Click the *Run Report* icon on the toolbar.**
   - **Select *Run...* from the *Reports* menu**

   The report you selected will appear in the Console. If the report requires parameters, you will be asked to provide these before the report appears. Parameters provide a way for you to filter the data that is returned on the report. If you do not specify a value for a parameter, that parameter will be ignored when filtering the data.

HOW TO PRINT A REPORT

In order to print a report, a printer must be installed on the Console machine.

1. **Expand the *Reports* node in the tree pane and select the report category that holds the report that you want to print.**
2. **Select the desired report in the data pane.**
3. **Click the *Print* button.**

HOW TO LAUNCH CRYSTAL DESIGNER FROM THE CONSOLE

If you own Crystal Reports version 9.0 or higher, you can design and modify reports from within the WinINSTALL Console.

***TIP:** WinINSTALL includes a set of HTML files which completely document the database schema, for both Oracle and SQL Server databases. To view these files, open* DBSchema.htm *in the* Bin\Help *folder on the share. You can use this information with third-party database mining tools and to create custom reports, if you have a copy of Crystal Designer.*

1. **Expand the *Reports* node in the tree pane.**
2. **Click the *Designer* button on the data pane.**
3. **If you have already specified the path to Crystal Designer, it opens. If you did not provide the path to Crystal Designer on the *Console Options* dialog, you will be prompted to do so now on the *Crystal Reports Path* dialog.**
4. **When Crystal Designer opens, you can create and modify reports from within WinINSTALL. After you have created and saved a report, you need to add it to the Console (see above, How to Add a Report to the Console).**

***NOTE:** In order for the* Designer *button to be active, you must own a licensed copy of Crystal Designer, Crystal Designer must be installed on the Console machine, and you must have configured the Console (on the Console Options dialog) to point to the Crystal Reports executable file on the Console machine.*

21

MISCELLANEOUS

This chapter discusses details of the WinINSTALL product which do not fit neatly into other chapters. The areas covered here include the following:

- WinINSTALL Logging
- WinINSTALL Utility Programs

WININSTALL LOGGING

WinINSTALL logging can have either or both of two targets: the WinINSTALL database and/or the Windows Event Log.

CONSOLE LOGGING

Console logging provides an audit trail of console user activity.

The only Console Logging setting (*View/Console Options*) is whether to log to the WinINSTALL database, the Windows Event Log, both, or neither.

When you click the *Console Log* node in the tree pane, you can view Console logging on either the *Database Log* or *Windows Event Log* tab in the data pane.

WININSTALL DATABASE LOG

When the *Console Log* node is highlighted in the tree pane, a list of all Console user activity that has been logged to the WinINSTALL database displays on the *Database Log* tab in the data pane.

You have the option of purging this log on demand.

The following information displays for each entry:

Type
- Because Console logging simply provides an audit trail of console user activity, all of the entries displayed on this tab are *Informational*.

Date
- The date on which the console user activity was logged to the WinINSTALL database.

Time
- The time at which the console user activity was logged to the WinINSTALL database.

Category • For WinINSTALL database logging, only entries in the *Console* category display.

Description • A brief explanation of the console user activity that was logged to the WinINSTALL database.

Event Code • Each Console activity has a numeric event code that corresponds to a particular activity. Refer to Console Log Event Codes and Descriptions for the meaning of each numeric code.

User • The name of the console user who was logged in when the activity was logged to the WinINSTALL database.

Click the *Purge* button to erase all of the existing entries that have been logged to the WinINSTALL database.

***TIP:** You configure WinINSTALL to log Console activity to the WinINSTALL database on the* Console Options *dialog, accessible from the* View/Console Options *menu item.*

WINDOWS EVENT LOG

When the *Console Log* node is highlighted in the tree pane, a list of all Console user activity that has been logged to the Windows Event Log on the Console machine displays on the *Windows Event Log* tab in the data pane.

You have the option of refreshing this view on demand.

The following information displays for each entry:

Type • Because Console logging simply provides an audit trail of console user activity, all of the entries displayed on this tab are *Informational*.

Date • the date on which the console user activity was logged to the Windows Event Log.

Time • This is the time at which the console user activity was logged to the Windows Event Log.

Category • For Windows Event Logging, only *Application* entries with the source of *WinINSTALL* for the console machine will display.

Description • This is a brief explanation of the console user activity that was logged to the Windows Event Log.

Event Code • Each Console activity has a numeric event code that corresponds to a particular activity. Refer to Console Log Event Codes and Descriptions for the meaning of each numeric code.

User • This is the name of the console user who was logged in when the activity was logged to the Windows Event Log.

Click the *Refresh* button to refresh this view on demand.

TIP: *You configure WinINSTALL to log Console activity to the Windows Event Log on the* Console Options *dialog, accessible from the* View/Console Options *menu item.*

CONSOLE LOG EVENT CODES AND DESCRIPTIONS

The *Event* column contains a numeric value which reflects the activity that was logged.

1	• the *** top-level list has been added.
2	• the *** top-level list has been removed.
3	• the *** list has been added to the +++ list.
4	• the *** list has been modified.
5	• the *** list has been removed from the +++ list.
6	• the *** package has been added to the +++ list.
7	• the *** package has been modified.
8	• the *** package has been removed from the +++ list.
20	• An error has occurred ***.
47	• Debug: ***.

WININSTALL UTILITY PROGRAMS

WinINSTALL includes several special purpose utilities located in the *bin* directory of the WinINSTALL share. These utilities are explained below.

CREATING A BASELINE FOR CONFLICT ASSESSMENT (WIBASELINEGEN.EXE)

This utility creates a "baseline" of a machine which can then be used for conflict assessment. A baseline defines the theoretical initial state of a machine before any packages are installed, such as the state of a machine with only a particular operating system and a service pack installed. You can include a baseline in a conflict assessment to determine if there will be potential conflicts when specific packages are installed on machines with that baseline. You run a conflict assessment baseline on a machine by remotely invoking *WIBaselineGen.exe* in the *bin* directory of a WinINSTALL share.

SCRIPT TO ADD A DATABASE USER (DBADDUSER.CMD)

This script enables an administrator to give permission to other users to use a Microsoft SQL Server 2000 or MSDE 2000 (Microsoft SQL Server 2000 Database Engine) database with NT Authentication. (NT Authentication is the WinINSTALL default for permissions on an MSDE 2000 or SQL Server 2000 database.)

This script must be run from a command prompt on a machine that has the OSQL.exe installed in the path. (OSQL.exe would normally be in the path on a machine where MSDE 2000 or SQL Server 2000 has been installed.) The script requires three parameters - the name of the database server/instance, the name of the database, and the name of the user to whom permission will be granted.

A sample command line is shown below:

```
dbAddUser.cmd MACHINE\ONDSQL WINSTALL8 MyDomain\MyUser
```

This script is intended to address situations where a Console user is unable to gain access to the database because of authentication problems.

Index

Symbols

.cub file 79, 81
.pcp file 76, 140

A

Access rights for packaging 33
Add a Database User 204
Add a new ASCII file to the list 183
Add a new ASCII file to the list. 183
Add a Report 198
Add existing package to Console 44
Add list to Console 44
Add new shortcut bar 30
Add Tab 153, 158
Add Tab (Adding INI Files) 180
Admin Subtab (Execute and User Interface Subtabs) 126
Administrative access rights 32
Advanced Options checkbox 53 to 55, 58, 61, 64, 67 to 69
Advanced Tab 111, 113 to 114, 116, 119, 121, 123, 189 to 190
Advertise Sub-tab 121, 127
Advertise Sub-tab (Execute and User Interface Sub-tabs) 127
Advertise Subtab (Execute and User Interface Subtabs) 127
Advertising 26, 50, 115, 127, 169 to 172
Advertising Packages for Distribution 170
Alphabetize contents of a list 45
AppID Tab 175
AppSearch Sub-tab 121, 128, 190 to 191
AppSearch Subtab 128, 191
ASCII Files Tab 182
Assembly Name Tab 177
Assembly Tab 176

B

Build a Package 48
Build Windows Installer Package Manually 73

C

CA File conflict subcategories and messages 88

CA INI File conflict subcategories and messages 89
CA Registry conflict subcategories and messages 88
CA Shortcut conflict subcategories and messages 89
Classes Tab 173
Classic Discover 51
Components Sub-tab 121, 124, 190
Components Subtab 124
Conditions Tab 111, 113
Conflict Assessment 25, 33, 83 to 84, 86 to 87, 197, 204
Conflict assessment access rights 33
Conflict Assessment node 25, 84, 86 to 87
Conflict assessment report 87
Conflict Assessment Wizard 83 to 84
Conflict Assessments tab 85
Console 20, 23 to 29, 31 to 32, 37, 43 to 45, 47 to 48, 52, 73 to 74, 79, 83 to 84, 87, 143, 153, 163, 170, 197 to 203
Console Layout 23
Console Log node 25, 201 to 202
Console logging 201
Console Security 31 to 33
Create baselines 83
Create new list 43
Crystal Designer 20, 25, 197, 200
Custom Action 137
Custom Action Sub-tab 131, 191
Custom Action Subtab 131, 191
Custom Action Wizard 133, 136
Custom Actions 121

D

Data Pane 23, 26, 28 to 29
Data Source Sub-tab 148
Data Source Subtab 148
Database 19, 84, 201
Database Schema 20
dbAddUser.cmd 204
Delete a Report 198
Delete Conflict Assessment 86
Delete shortcut from shortcut bar 31
Dialogs Sub-tab 121, 129, 190 to 191
Dialogs Subtab 129, 191
Digital Signature 92
Directory Tab 121 to 122, 189, 193
Discover Advanced Options 55
Discover Wizard 37, 47 to 48, 52 to 53, 55, 58, 60 to 61, 63, 67 to 68, 71, 95, 187
drag and drop 27
Driver Sub-tab 149

Driver Subtab 149

E

Edit a Package 49
Edit an ASCII file that was added to the list. 184
Edit Conflict Assessment 86
Edit name of shortcut bar 31
Environment Tab 185
Extensions Tab 174

F

Favor Advertising 170

G

General Information for Installing Windows Installer Component 116
General Information for Installing Windows Installer Feature 111
General Information for Installing Windows Installer Package 116
Generate Baseline 84
Generate Conflict Assessment Baseline 84

I

ICE message 80
ICEs 79, 81
Ignore Tables Sub-tab 192
Ignore Tables Subtab 192
inheritance icons 46
INI Files Tab 180
Install Modes Tab 116, 121 to 122
Install sub-tab 125
Install Sub-tab (Execute and User Interface Sub-tabs) 125
Install Subtab (Execute and User Interface Subtabs) 125
Installation 123
Internal consistency evaluators (ICEs) 79

J

Just-in-time installation 169

L

Launch 33, 101, 121, 123, 125, 190, 197, 200
Launch Condition Sub-tab 121, 125, 190
Launch Condition Subtab 125, 190
Launch Crystal Designer 197, 200
List Pane 23, 25
List pane 25
Lists 32, 37, 43, 163

M

Main CA Conflict Categories 88
Menubar 24
Merge Modules Tab 191
Move shortcut bar 30
MSI Custom Action 137
MSI Custom Action Wizard 132
MSI Custom Actions Subtab 131
MSI Package Validator dialog 81
MSI packages 47, 131, 148, 187
MSI Table Editor 76 to 77, 121, 140

P

Package History 121
patch creation package 76, 140
Patch Wizard 37, 48, 75 to 76
Patches 43
Print a Report 199
ProgID Tab 174

R

Recommended WinINSTALL Workflow 14
Reference Machine 20, 52
Reference Machine WinINSTALL Database 52
Registry 25, 49, 88, 108, 157 to 160
Remove list from Console 44
Remove package from Console 44
Remove shortcut bar 30
Remove Tab 145, 159
Rename shortcut 31
Reorder contents of a list 45
Repairing Validated Windows Installer Package 81
Report 79 to 80, 87, 198 to 199
Reports 25, 29, 32, 87, 197 to 200
Reports node 25, 29, 87, 197 to 200
Reports tab 197
Reserve Cost Tab 118
Run a Report 199
Run Conflict Assessment 86

S

Schema 20
Select Directory Dialog 99
Sequence Sub-tab 121, 125, 192
Sequence Subtab 125, 192

Shortcuts Bar 24, 27, 30 to 31
Show/hide the shortcuts bar 30
Signing an External Cabinet File 93
Software Distribution 24 to 25, 28, 32 to 33, 37, 43 to 45, 48 to 49, 73 to 74, 80, 122, 170, 172, 187
Software Distribution Lists 24, 28, 32, 37, 43, 48 to 49, 73, 80, 122
Software Distribution Lists node 24, 28, 32
Software Distribution node 24 to 25, 28, 33, 37, 43 to 45, 48 to 49, 73 to 74, 80, 170
Specify an INI file to be added 181
Specify INI file section to add 182
Start Page 24
Summary Sub-tab 107
Summary Tab 107, 111 to 112, 116, 189

T

Time 201 to 202
Toolbar 24
Translator Sub-tab 150
Translator Subtab 150
Tree Pane 23 to 25
TypeLib Tab 172

V

Validating Windows Installer Package 80
Validation messages 79
View Results of Conflict Assessment 87
View/Edit Tables 140

W

What's New 14
WIBaselineGen.exe 84, 204
Windows Installer package 25, 37, 47 to 49, 76, 79 to 81, 91, 101, 107, 111 to 112, 116, 121 to 125, 129, 143, 145 to 150, 153, 158 to 160, 163 to 164, 166 to 167, 170 to 172, 180, 185, 187 to 188
WinInstall Conflict Assessment Baseline Generator dialog 84
WinINSTALL Conflict Assessment Wizard 84
WinINSTALL Console 19, 23, 29, 31, 47, 52, 75, 200
WinINSTALL Database 19, 201
WinINSTALL database 25, 84, 197, 201 to 202
WinINSTALL package 95, 121, 180 to 181, 183
WinINSTALL Product Family 14
WinINSTALL Share 19
WinINSTALL share 52, 83 to 84, 203 to 204
Work with Lists 47
Work with Windows Installer Packages 47