

DESKTOP AUTHORITY
MSI STUDIO™



Desktop Authority MSI Studio
Administrator's Guide

SCRIPTLOGIC

**Copyright © 1998-2007 ScriptLogic Corporation and its licensors. All rights reserved.
Protected by U.S. Patent 6,871,221 with other U.S. and International Patents Pending.**

This publication is protected by copyright and all rights are reserved by ScriptLogic Corporation. It may not, in whole or part, be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine-readable form without prior consent, in writing, from ScriptLogic Corporation. This publication supports MSI Studio 3.5. It is possible that it may contain technical or typographical errors. ScriptLogic Corporation provides this publication “as is,” without warranty of any kind, either expressed or implied.

ScriptLogic Corporation
6000 Broken Sound Parkway NW
Boca Raton, Florida 33487-2742

1.561.886.2400
www.scriptlogic.com

Trademark Acknowledgements:

MSI Studio, Desktop Authority, ScriptLogic and the ScriptLogic logo are either registered trademarks or trademarks of ScriptLogic Corporation in the United States and/or other countries. The names of other companies and products mentioned herein may be the trademarks of their respective owners.

DOCUMENTATION CONVENTIONS

Typeface Conventions

Bold Indicates a button, menu selection, tab, dialog box title, text to type, selections from drop-down lists, or prompts on a dialog box.

CONTACTING SCRIPTLOGIC

ScriptLogic may be contacted about any questions, problems or concerns you might have at:



ScriptLogic Corporation
6000 Broken Sound Parkway NW
Boca Raton, Florida 33487-2742



561.886.2400 Sales and General Inquiries
561.886.2450 Technical Support



561.886.2499 Fax



www.scriptlogic.com

SCRIPTLOGIC ON THE WEB

ScriptLogic can be found on the web at www.scriptlogic.com. Our web site offers customers a variety of information:

- Download product updates, patches and/or evaluation products.
- Locate product information and technical details.
- Find out about Product Pricing.
- Search the Knowledge Base for Technical Notes containing an extensive collection of technical articles, troubleshooting tips and white papers.
- Search Frequently Asked Questions, for the answers to the most common non-technical issues.
- Participate in Discussion Forums to discuss problems or ideas with other users and ScriptLogic representatives.

Table Of Contents

INSTALLATION	1
SYSTEM REQUIREMENTS	1
INSTALLATION	3
REGISTRATION AND EVALUATION.....	7
APPLICATION PACKAGING CONCEPTS.....	9
WHAT IS PACKAGING SOFTWARE?	9
WHAT IS AN MSI FILE?	9
WHAT IS MICROSOFT WINDOWS INSTALLER?.....	9
WHAT IS MSI STUDIO™?.....	10
TYPES OF VENDOR SUPPLIED SOFTWARE INSTALLATIONS	12
THE MSI STUDIO PACKAGING PROCESS.....	13
USER INTERFACE.....	15
MENU BAR	15
TOOLBAR	17
USING THE INSTALLER DESIGN STUDIO WIZARD	19
USING THE PACKAGING WIZARD	20
USING THE PACKAGING WIZARD	20
REPACKAGE AN APPLICATION FOR DEPLOYMENT	22
NEW REPACKAGE BY MONITOR AND SNAPSHOT	24
MODIFY AN EXISTING MSI	28
TEST THE INSTALLATION.....	30
CREATE A WRAPPER INSTALLATION	31
CREATE NEW SETUP INSTALLATION	32
OPEN EXISTING PROJECT	33
INSIDE THE MSI	34
TABLES	34
IQ VIEWS	35
<i>Overview.....</i>	<i>35</i>
<i>Project Details.....</i>	<i>36</i>
<i>Add or Remove Programs.....</i>	<i>38</i>
<i>Upgrades</i>	<i>40</i>
<i>Application Requirements.....</i>	<i>42</i>
<i>Project Organization.....</i>	<i>46</i>
<i>Files and Registry.....</i>	<i>50</i>
<i>Shortcuts and Other Items</i>	<i>60</i>
<i>Dialogs and Actions</i>	<i>72</i>
<i>Recycle Bin</i>	<i>11</i>
VALIDATION.....	12
VALIDATION.....	12
<i>MCE Validation.....</i>	<i>12</i>
<i>ICE Validation.....</i>	<i>12</i>

TOOLS	14
MSI STUDIO TOOLS	14
<i>What is a Transform?</i>	14
<i>Create a New Transform</i>	15
<i>Create a Response Transform</i>	17
<i>Editing an Existing MST</i>	18
MSI STUDIO RUN WIZARD	19
UPGRADE A PROJECT	23
<i>One off Upgrade</i>	24
<i>Automatic Upgrades</i>	24
EXTRACT COM FILE FILES	25
CONVERT REGISTRY COM	26
SEARCH FOR MODULES	28
CREATE MERGE MODULES	29
EXTRACT CABS	31
VIEW THE DIFFERENCE BETWEEN 2 MSIS	33
SOFTWARE WAREHOUSE MANAGER	35
OPTIONS	37
<i>Template Options</i>	37
<i>Merge Modules</i>	38
<i>Software Warehouse</i>	39
<i>Other Options</i>	40
<i>Repackaging</i>	41
<i>Build Options</i>	43
<i>Digital Signature Option</i>	44
SOFTWARE WAREHOUSE	45
OVERVIEW	45
<i>Using the Software Warehouse</i>	47
<i>Query Conflicts</i>	55
REFERENCE	58
COMMAND LINE OPTIONS	58
GLOSSARY	59
INDEX	61

Installation

SYSTEM REQUIREMENTS

Desktop Authority MSI Studio can be installed on either a server or workstation. .NET Framework 1.1 is required to install and run the application.

The following applications are necessary for proper operation of MSI Studio. These will be installed by the MSI Studio installer if they do not exist on the installation machine.

System Requirements

- .NET Framework 1.1

Professional Edition

System Requirements (installed by MSI Studio, if necessary)

- MDAC v2.80 or higher
- Windows Installer 3.0 or higher
- SQLDMO (SQL Server 2005 Backward Compatibility)
- MSDE, SQL Server Express or SQL Server 2000/2005

OS Requirements - Package Management

- Windows 2000 SP4 or above
- Windows 2003 SP1 or above
- Windows XP SP2 or above
- Windows Vista

OS Requirements - Package Deployment

Any version of Windows that support Windows Installer (95/98/Me/NT/2000/XP/2003/Vista)

Standard Edition

OS Requirements - Package Management

- Windows 2000
- Windows 2003
- Windows XP
- Windows Vista

OS Requirements - Package Deployment

Any version of Windows that support Windows Installer
(95/98/Me/NT/2000/XP/2003/Vista)

INSTALLATION



The following series of steps walk through the installation of Desktop Authority MSI Studio.

Be sure to read the System Requirements to make sure your systems meet the minimum requirements and learn about the information you will need to know for the setup.

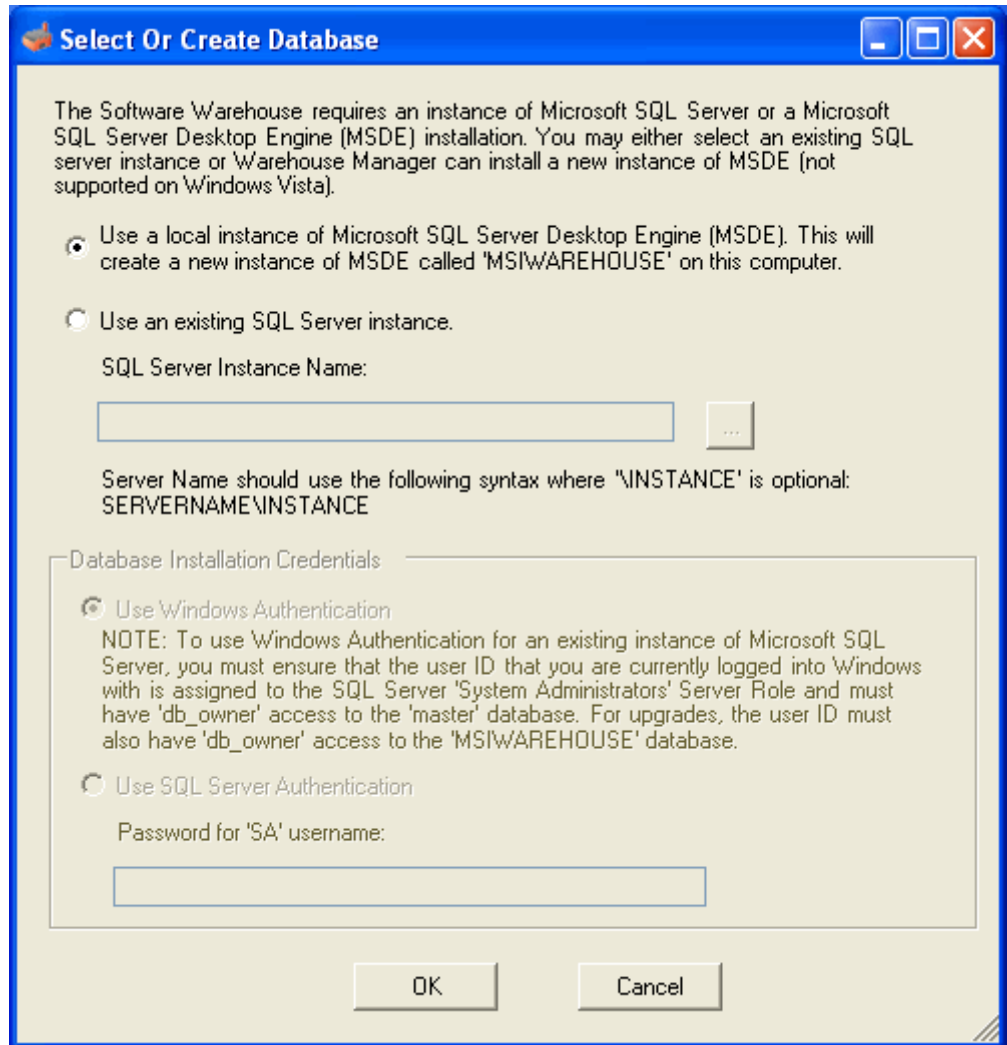
The installation wizard walks through a series of dialog boxes prompting for information that is needed to install MSI Studio to your computer. Click **Next** on each dialog box to advance to the next option. Click **Back** to go to the prior dialog box. Click **Cancel** to abort the install.

1. **Welcome** is the initial dialog box. Click **Next** to continue.
2. The **License Agreement** dialog box appears. If you agree with the license agreement, click **Accept**. Click **Next** to continue.
3. **(MSI Studio Professional Edition only)** The Software Warehouse is comprised of a web service component and an SQL database instance. The database instance can use either Microsoft SQL Server Desktop Engine or Microsoft SQL Server. Select **Install Client Only** to install MSI Studio without the database installation. This option should be used when an existing SQL server instance will be used for the Software Warehouse or if a Warehouse database already exists. Select **Install the Client and Server** to install both MSI Studio along with a database instance. This selection will install a new instance of MSDE.
4. On the **Destination Directory** dialog box, select a path and destination folder. The default installation path is *x:\Program Files\ScriptLogic Corporation\MSI Studio*. Press the **Browse** button to select a different path. Click **Next** to continue.
5. Click **Install** on the final installation dialog to proceed with the installation. Once the file copying portion of the install is complete, click **Finish**. Proceed to the following configuration steps for MSI Studio Professional Edition.

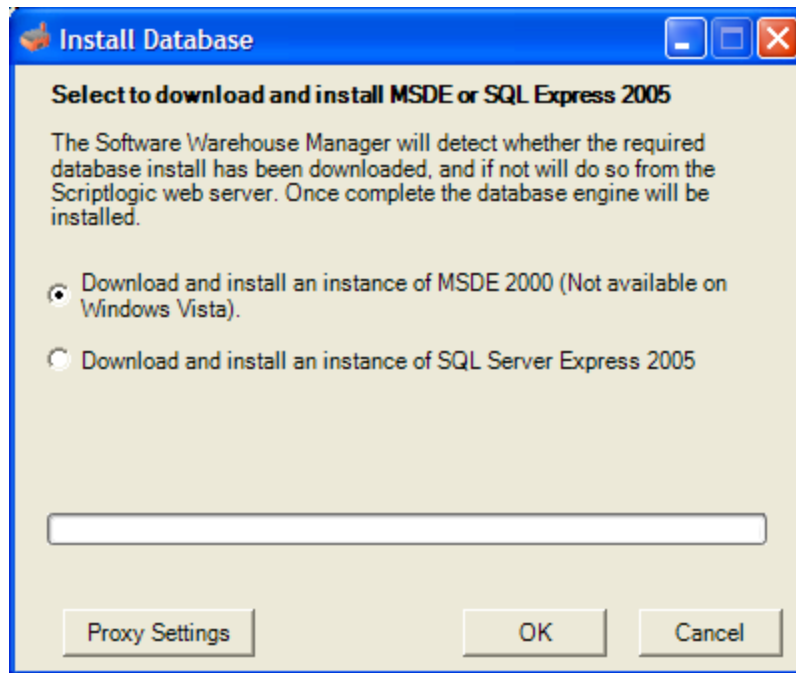
The following steps apply to MSI Studio Professional Edition only.

6. Desktop Authority requires Microsoft Data Access Components (MDAC) to function properly. If the version of MDAC on the installation computer is not current, you will be prompted to upgrade to MDAC 2.8. Click Yes to upgrade to MDAC 2.8. Click No to skip this step and install it at a later time via the Software Warehouse Manager. The system will automatically restart after MDAC 2.8 is installed.

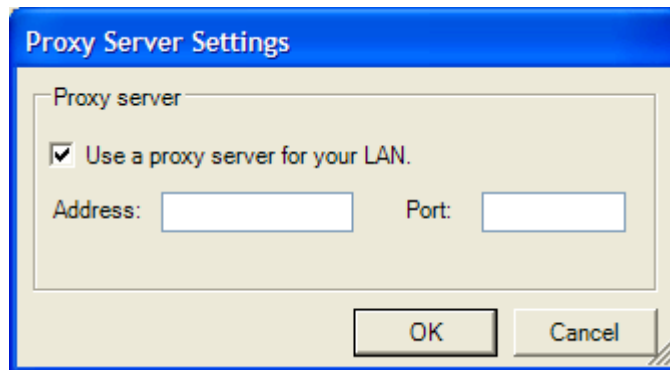
7. Another system requirement of the MSI Studio Warehouse Manager, is the presence of SQL DMO. When prompted, click Yes to install SQL DMO. Click No to skip this step.
8. The next step of the MSI Studio install is the configuration of the database for the Software Warehouse Manager. Click Yes to configure it now; click No to configure it at a later time.
9. MSI Studio can install a new instance of MSDE or use an existing SQL Server instance. Select an option in the dialog.



If Install the Client and Server is selected, a dialog box will be displayed to prompt for the version of SQL server to download and install.



If your internet connection requires a proxy server. Click Proxy Settings and configure the proxy server settings.



Click OK to continue.

The selected database engine will be downloaded and installed automatically.


Note: MSDE is not supported on Vista environments. SQL Server Express 2005 should be downloaded and installed on Vista.

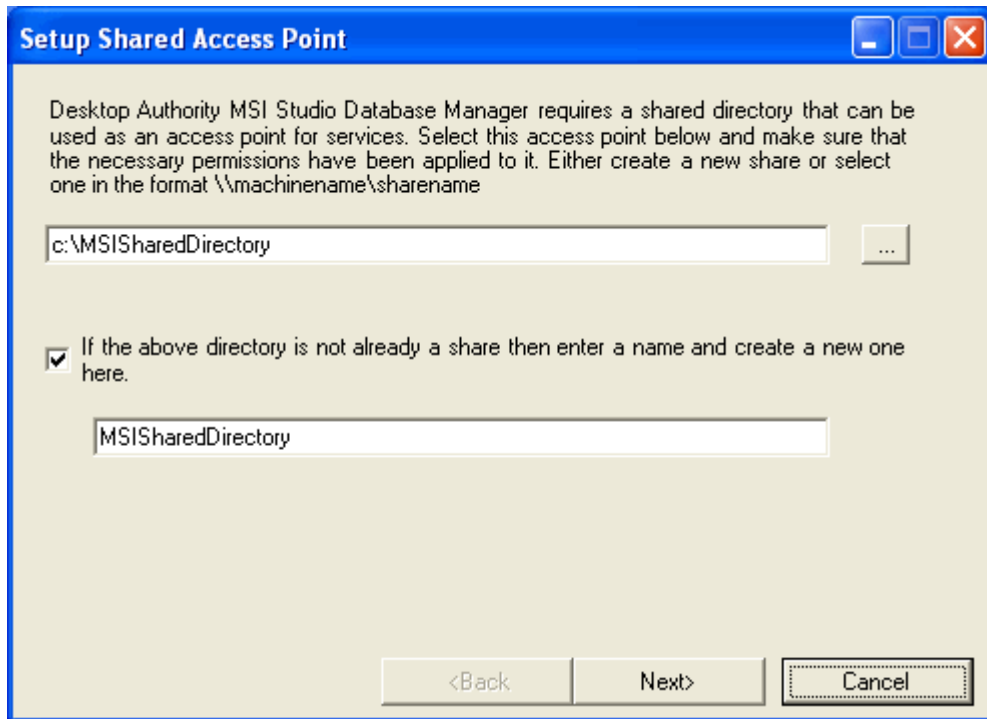
When choosing to use an existing SQL Server instance, type in the SERVERNAME\INSTANCE, where \INSTANCE is optional or press



to select from a list of existing SQL instances. Select an authentication method for SQL Server.

Click OK to continue and install the database. Click Cancel to skip this step.

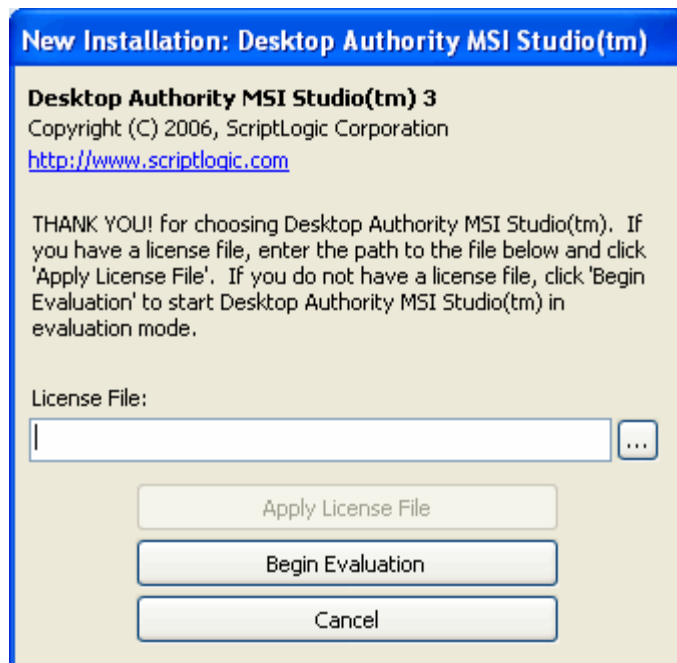
10. Finally, the MSI Studio install will create a shared access point. Users must have READ + WRITE access to this folder. Enter the share name in the form of \\machinename\sharename. Click  to browse out and select an existing share. If a new share must be created, select the checkbox to created it and enter the share name in the entry provided.



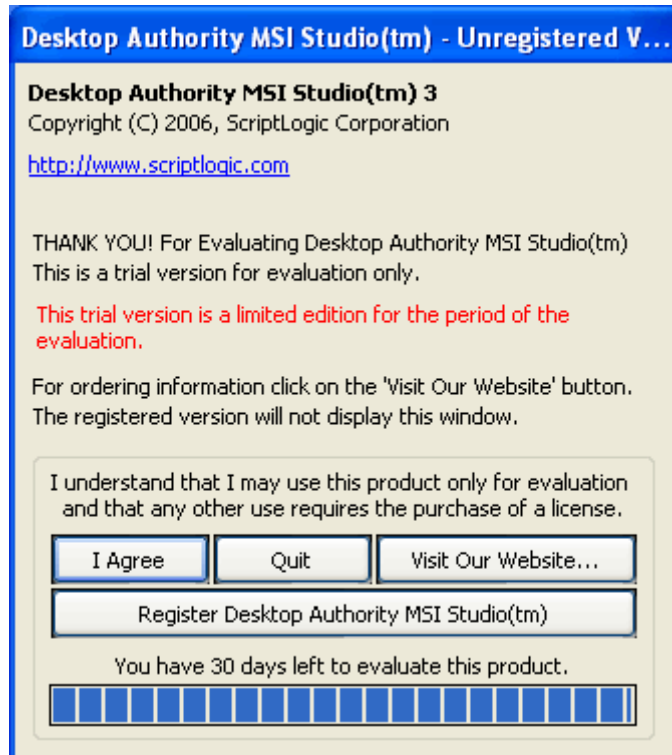
11. At the completion of the install the Software Warehouse Manager is displayed. The Manager will display the status of the web service, database installation and shared access point.

REGISTRATION AND EVALUATION

The first time Desktop Authority MSI Studio is run following installation, you are prompted to register the product. If you will be evaluating MSI Studio, click **Begin Evaluation**. If you have purchased MSI Studio and have been given a license file, enter the path and filename to the license file. Alternatively, click to browse to the license file. Once the license file is entered, click **Apply License File**.



The evaluation period is 30 days. To begin the evaluation, click **I Agree**. This will provide your agreement to the terms and usage of the evaluation software. Click **Quit** if you do not agree with the evaluation usage. The software can be registered from this dialog by clicking **Register Desktop Authority MSI Studio**.



To register MSI Studio at a later time, select **Help > About Desktop Authority MSI Studio** from the menu bar. Click **Apply License File** to register your software.



Application Packaging Concepts

WHAT IS PACKAGING SOFTWARE?

As part of a comprehensive desktop life cycle management strategy, organizations need to centralize, automate and simplify the task of standardizing the deployment of applications. To establish consistent results across the enterprise, IT organizations employ packaging solutions to standardize application installations. Application packagers need a powerful, centralized solution that also simplifies the creation, editing and management of Windows Installer (MSI) files. These solutions come under the broad category of Packaging Software.

Traditional software installers come in many formats, many are script based, some are executable code, and others just come with a set of manual instructions of where to copy files and how to register them. The problem with having a variety of installation methods is that there is often no way of knowing what changes are being made to a system, no way of monitoring potential conflicts between applications, and no way to repair an application if it is in an unstable state. It is common with legacy installations for one application to place another in an unstable state by modifying a component to work with a newer version of the software. Packaging Software helps to create a stable installation that is customized for an Enterprise's environment.

WHAT IS AN MSI FILE?

Simply put, an MSI file is a Microsoft Windows Installer file. MSI files contain all of the information Windows Installer requires to install or uninstall an application. Contained in the MSI file is the installation database, data streams, transform files, source files or CABs that are required by the installation. Transform files provide data to the installer that an end user would normally be prompted for while installing an application. Microsoft's Windows Installer consumes MSI files and installs the application to the desired computer.

WHAT IS MICROSOFT WINDOWS INSTALLER?

Microsoft Windows Installer is a service that installs, repairs, or removes software according to instructions contained in the Installer's package file (.MSI). Microsoft Windows Installer is an application that is shipped with all Windows Vista, Windows 2003 Server, Windows XP, Windows 2000 and Windows Me. It is also available as a service pack for Windows NT 4.0, Windows 98 and Windows 95.

WHAT IS MSI STUDIO™?

MSI Studio™ is a complete software packaging solution used by system administrators, software packagers and setup developers worldwide. It has a complete set of features used to create Windows Installer (MSI) files from legacy software installations, tools to modify existing Windows Installer packages using transforms, conflict detection and resolution and pre-flight testing of any package.

- **Repackage**

Create custom MSIs from both legacy and existing MSI-based installs.

- **Edit**

Modify MSIs using either a graphical view or by direct access to MSI tables.

- **Manage**

Improve your MSIs with advanced conversion, merging, validation and extraction features.

- **Build**

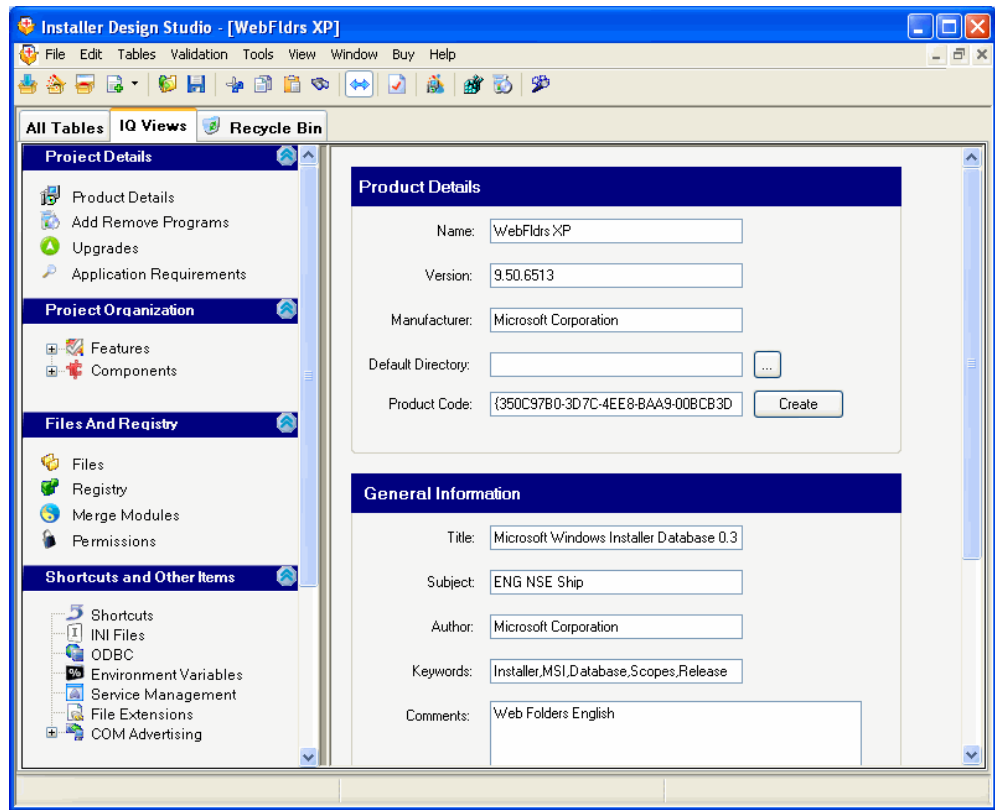
Create MSI projects using cab files, compressed MSIs, bootstrapped MSIs and certificates.

- **Conflict Detection**

Detect conflicting settings within Installer packages or groups of packages.

- **Pre-flight Testing**

Test the install of a package before it is deployed to network users.



TYPES OF VENDOR SUPPLIED SOFTWARE INSTALLATIONS

Software installations can generally be grouped into one of three categories for packaging purposes:

Legacy Software Installer

Lots of software is distributed in a non-MSI format. The formats include, but are not limited to, script execution, executable files, such as SETUP.EXE, or manual instructions providing the steps necessary to manually copy the files to the proper folders.

These types of software installations should be repackaged into a Windows Installer (MSI) format. Essentially, the changes that are made to the system during the original installation process will be captured and a new Windows Installer-based installation package will be created for the application.

Existing Windows Installer Software Package

As software vendors come to understand the benefits of using Windows Installer-based installations, more of them are distributing their software as .MSI files (or a .MSI file wrapped in a .EXE).

It is generally accepted that vendor-supplied Windows Installer packages should not be modified. There are two main reasons why. First, it's a waste of time repackaging something back into its original format. Second, if you repackage the original MSI file, any updates or patches supplied by the vendor will also need to be repackaged. This can lead to problems further down the road.

However, it is possible to modify an .MSI file using a *transform*. A transform is a type of Windows Installer database that is merged with the MSI file at runtime. The main reason transform files are used is to create *responses* to options given during the UI sequence of an install, such as supplying the serial number and selecting the features you want installed so the installation can be run silently.

Installations that cannot be repackaged in MSI Format

This includes such things as driver installations, Windows service packs, updates and hotfixes. Hardware changes should not be captured by a repackaging process. These types of installations work based on specific machine hardware. Windows Updates should not be repackaged as they often change protected files which Windows Installer cannot access.

In cases like these it is often easier to wrap the executable or vendor supplied installation in an .MSI file to make deployment easier.

THE MSI STUDIO PACKAGING PROCESS

Step 1. Test Your Application Before Starting

The first step, and one which is often over looked, is to test the software before you repackage it. The aim of this step is to make sure that it works on your system. If it does not work then there is not much point in repackaging it. It is also worth checking with the vendor to see if there is a more recent release of the application.

Install the application using the vendors installation procedure and make sure that it works. While installing, note the method of installation (check to see if it is using Windows Installer, see step 2), and the steps involved in configuring it.

Step 2. Figure out the Method of Installation

Decide what type of installation the application uses. This is necessary in order to decide upon the repackaging method to use. The basic software installations use a legacy software installer program, an existing Windows Installer package or an application that includes driver updates or Windows service packs, updates and hotfixes, as discussed in the Vendor Supplied Software Installations section.

The first step in deciding this is to see whether it is a Windows Installer installation.

Check the CD or installation source for signs of an MSI file. Often, an installation may come with an MSI file and an executable, which might serve no other purpose than to check that the Windows Installer runtimes are present.

If there is no MSI file but there is an executable, you should also check that the MSI is not compressed inside the executable. To do this you can run the executable and check the files in the temporary directory. Often an MSI file and other related files might be uncompressed into the temp directory. Also, check Windows Task Manager to see whether an extra MSIEXEC.EXE process is started or not. This is another sign that the installation is using a Windows Installer package.

If it is not Windows Installer based you will need to check whether it can be repackaged or not. Often it is not possible to determine this until well into the repackaging process but earlier checks can save you time. Windows Updates often update protected system files and therefore cannot be repackaged. Legacy driver installations are also another example of an application which can cause problems when repackaging. Although you should try and repackage everything into MSI format just be aware that this is not always possible in which case you might want to 'wrap' the applications installation in an MSI file.

Step 3. Follow the Appropriate Process

- Repackage a legacy installation (No MSI file)
- Modify an Existing MSI File
- Create a Wrapper Installation

Step 4. Conflict Detection and Resolution

***The Software Warehouse is only available in the Professional version of MSI Studio**

Once the MSI file has been created and tested, it should be imported into the Software Warehouse. The Software Warehouse provides valuable information regarding conflicts between the newly created package and existing packages or groups of packages that it may be installed with. Following any conflicts, the package may be modified with any necessary corrections.

Repackaging Tips

When repackaging it is recommended that you have at least 3 machines that can easily be rebuilt with a clean build image. This can be achieved by having 3 physical machines and rebuilding them with a utility such as Ghost, or by using virtual machines. One machine can be used as your development machine, one as a test machine and one as a clean machine to do the repackaging on.

It is important to keep the Repackaging machine clean by rebuilding it between capturing changes made by an application. You will need administrator privileges on the machine reserved for Repackaging and will need to be able to access the main drives either as a share (C) or admin share (C\$). You will also need permission to launch remote applications through WMI (Windows Management Instrumentation).

When repackaging be sure that you are logged onto the Development and Repackaging machine using the same username. When testing on the Test machine, make sure you test with different users with limited privileges.

User Interface




















MENU BAR







The Menu bar provides various menus used to perform packaging operations. Menu items are enabled or disabled according to the selected packaging function. Disabled menu items appear on the menu but are grayed out and will not perform any action if clicked on.

Menu	Description
File	Use the File menu to create New , Open existing, Delete and Save packages. This menu also provides links to clean the IDS database and export a package to XML.
Edit	This menu includes items for Undo, Cut, Copy, Paste, Copy Rows, Paste Rows, Delete Rows, Select Rows, Find and Replace options and package summary information.
Tables	Use the Table menu to Add, Drop, Import and Export Tables.
Validation	Use the Validation menu for MCE and ICE package validation.
Tools	The Tools menu provides several useful tools such as creating a transform file, repackage a project by snapshot, launch the MSI with the Run Wizard, Upgrade a project, extract COM files from project, extract CAB files, compare MSIs, test for conflicts and others.
Warehouse *The Software Warehouse is only available in the Professional Version of MSI Studio.	Open the Software Warehouse for Conflict Detection and Resolution.

View	The View menu provides the ability to open the Output panel.
Window	Use the Window menu to change the display of open windows when many windows are open at the same time. Choose to Tile Horizontally, Tile Vertically or Cascade. This menu can also be used to bring an open window into focus.
Buy	Purchase ScriptLogic products online.
Help	Use the Help menu to get system help, make a ScriptLogic purchase or enter the IDS license key. From the help menu, check for product updates. The Help menu also displays IDS system information.

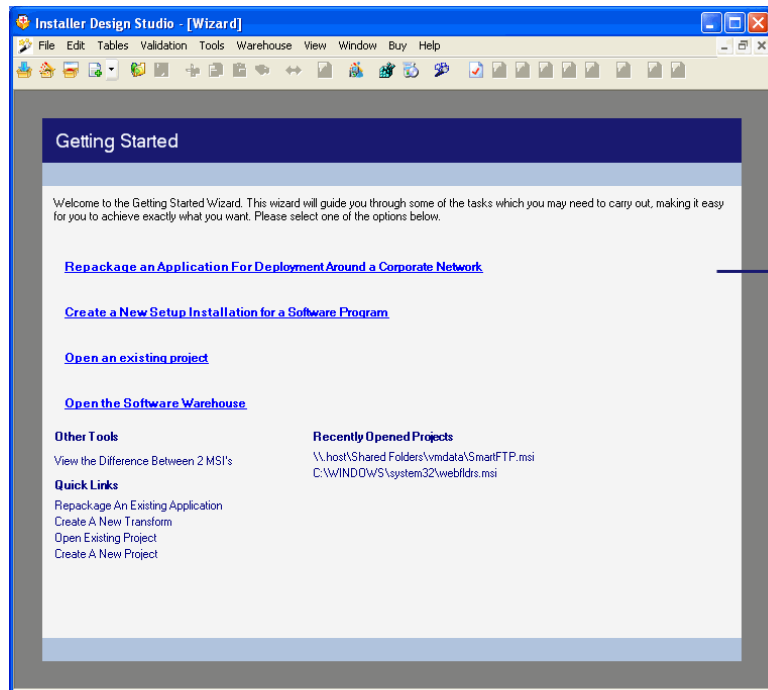
TOOLBAR

	Repackage a legacy application
	Create a transform for an existing MSI file
	Wrap an executable file in a MSI
	Create a new MSI file from a template
	Open an existing Windows Installer file
	Save project
	Cut the contents of current cell
	Copy the contents of the current cell
	Paste from the clipboard to the current cell
	Searches the tables for text
	Resize the columns of the table view, either to largest value or so all fit
	Run the MCE Validation Wizard
	Launch the Installer Design Studio Run Wizard
	Launch Regedit
	Launch Add/Remove Programs dialog
	Display the Installer Design Studio Wizard
	Open the Software Warehouse
	Import a project into the Software Warehouse
	Add New Application

	Add New Group
	Add existing project to an application
	Add an application to a group
	Query Conflicts
	Connect and refresh view
	Recalculate all conflicts

USING THE INSTALLER DESIGN STUDIO WIZARD

The Installer Design Studio wizard interface guides the user through the process of repackaging or creating a new MSI package or opening an existing MSI package.



Wizard Interface

*The Software warehouse is only available in the Professional Version of MSI Studio

Using the Packaging Wizard

USING THE PACKAGING WIZARD

The packaging wizard guides the packaging process by supplying the necessary steps to create a Windows Installer package.

The first page of the Packaging Wizard is **Getting Started**. The Getting Started page provides a means to repackage existing applications, create new packages or open existing packages.

Repackage an Application for Deployment around a Corporate Network

Select this option to be guided through the process of repackaging an application for the corporate environment. You will be given the option of repackaging a legacy installation or to create transforms for an existing MSI file.

Create a New Setup Installation for a Software Program

This option will guide you through the process of creating a new software installation program. Select this option if you have a commercial software application which needs an installation created for it. This is also useful if you have a file of configuration settings which you repeatedly need to install.

This option will provide the option to select a template. If you wish to create a template of your own, create a base MSI file and drop it into the template folder. Templates typically contain dialog boxes and a UI theme of your choice.

Open an existing project

Select this option to open and edit a project that already exists. You can open MSI files, transforms (MST's) or other compatible files such as WSE's and ISM's.

Open the Software Warehouse

Open the Software Warehouse for Conflict Detection and Resolution between selected MSI files.

Other Tools

This option contains links to other tools useful to MSI projects.

Quick Links

A linkable list of frequently used MSI Studio features.

Recently Opened Projects

A linkable list of projects that have recently been opened in MSI Studio. Clicking on a project will open the package for editing.

REPACKAGE AN APPLICATION FOR DEPLOYMENT

Repackage an Existing Application

Before making a selection, decide on the process needed to repackage the application. Use the following steps to guide in this selection:

- **Decide whether the existing package is in MSI format.**

Look at the source installation and see if there is a file with an MIS extension. If the installation application itself is not an MSI, there could still be an MSI file which has been wrapped inside the executable. To determine if there is an MSI file within the executable, run the executable and check the temp directory to see if any MSI files have been unpacked. Another way is to determine if there is an MSI file within the installation is to check the machine's running processes for an extra MSIEXEC.EXE process. MSIEXEC.EXE is used to execute MSI files.

- **If there is no existing MSI file, check to see if the application to be deployed is a Windows patch or hardware driver.**

If the installation program is a patch or contains hardware changes, then it is possible that the change can not be captured (hardware changes can not always be captured and patches often update protected Windows files).

Having determined if an MSI file is available, you can now make a decision on which process to take.

- **Repackage a legacy installation (NO MSI file)**

Select this option if the installation is not Windows Installer based and does not contain Windows patches, hotfixes, device drivers or network protocols. The new MSI installation will be created by installing the software to a clean machine that is being monitored. Snapshots of the machine are made before and after the installation. From the difference in these snapshots, the new MSI file is created.

- **Modify an existing MSI file (you have an existing Windows Installer installation)**

Select this option if you have found an existing MSI as part of the vendor supplied installation package. Vendor supplied MSI files should not be repackaged into a new MSI file. Instead, a transform file should be created to add or modify any configuration options as well as allow it to run silently.

Generally speaking, a vendor-supplied MSI file should not be modified. The reason for this is that future upgrades or patches supplied from the software manufacturer will not run on your re-created MSI file. Therefore, you will be stuck repackaging all future releases as well as the initial one. This can create a lot of unnecessary work.

- **Test the installation for MSI files and extract them if present**

Select this option to extract MSI files that are embedded within a setup executable.


- **Wrap an Executable in an MSI**

Select this option for installations that cannot be repackaged into an MSI file or with a Transform. A Wrapper is an MSI file that envelops the installation files and will execute the setup file when the MSI file is executed. Service packs, hotfixes, device drivers and network protocols are examples of installations that might require a wrapper install.

NEW REPACKAGE BY MONITOR AND SNAPSHOT

To repackage a non-Windows Installer based installation, a new MSI installation will be created by installing the software to a clean machine that is being monitored. Snapshots of the machine are made before and after the installation. From the difference in these snapshots, the new MSI file is created.

WHAT

Provide the full path and file name of the MSI file you wish to create. Press  to browse to a path and specify a new file name. The file must have an MSI file extension. This MSI file will be created at the end of the repackaging process.

WHERE

The changes that need to be captured can be carried out on the local computer or on another computer that requires no client installation.

It is recommended that you work from a development machine but connect to another which can be rebuilt to a clean state, i.e. it only contains your base build with no other application installed.

Run the installation on the local machine

Select this option to install the application to and monitor the local machine.

Run the installation on a foreign machine

Select this option to install the application to and monitor a foreign machine.

Installer Design Studio uses the Windows Management Instrumentation (WMI) infrastructure to detect the changes made on foreign machines. Use the following guidelines If you are having trouble connecting to the foreign machine:

- Make sure that you are logged on as a local administrator, using the same username and password. Test this by trying to connect to the foreign machine's admin share,

```
net use \\<foreignmachine>\c$
```

where <foreignmachine> is the name of the foreign machine you want to use.

- The password should not be blank.
- Make sure both machines are on the same domain. You can use the command `net use *` on a console to see the computers that are connected to the domain. Both machines should appear on the list.
- Make sure that `reg.exe` or `regedit.exe` are in the system path. For Windows 2000 and Windows Server 2003 you might need to copy `regedit.exe` to the system directory.
- It is a good idea to share the drives that are to be monitored for changes.
- Configure firewalls or turn them off
- Be sure that you are able to run remote applications through WMI.

Select or enter the name of the machine you wish to monitor

Select the foreign machine to monitor from the drop list.

Test

Click Test to confirm the connection to the remote computer. The remote computer must be able to be successfully connected to.

Username

Enter the Username of the user to log on to the remote machine as.

Password

Enter the Password of the user to log on to the remote machine as.

HOW

The repackaging process will first perform an initial scan of the specified computer, local or foreign. This initial scan is used as a comparison after the application is installed. It is compared to an after scan of the computer to determine the changes that were made by the installation.

Select whether to do a first up scan or to use previous captured information

Scan the hard drives and registry initially

Select this option to perform an initial scan during the repackaging process. It is a good idea to do an initial scan of a clean machine. This initial scan can then be used each time the machine is rebuilt back to a clean state.

Use the information from the previous setup capture run

Select this option if a prior scan is available.

Advanced Options

Search and add merge modules after capture

A merge module provides a standard way to deliver specific components to a machine. They are used to ensure that the proper version of a component is installed with its related files, resources, registry entries and setup logic. Select this box if merge files will be used during this repackaging process.

Display the Remove Captured Information dialog

Select this box to provide the ability to remove some of the captured information before the final MSI file is created.

Create advertising entries from the registry entries

MSI files have the ability to display perform advertising. Advertising is the process by which not all features included in the package are installed. However, the first time the user attempts to use one of these uninstalled features, it will automatically be installed onto the system.

Select this box to process the system's registry entries for any advertising associated with this application.

Copy source files to repository

It is a good idea to copy all source files to a repository so that the package can be easily rebuilt, if necessary. If the source files are not available, the application will need to be reinstalled and reprocessed.

Select this box and enter or browse to the repository path.

Repackage using file system monitoring

There are two ways to figure out which files have been modified or added to the system by an install. By completing a full initial scan and then a full scan at the end of the install, and detect the changes, or use the file system monitor which will detect changes as they occur.

The file system monitor is faster but sometimes it might be necessary to do a full before and after scan. Select this box to use the File System Monitor.

Test for conflicts while repackaging

Select this box to run the Conflict Builder when repackaging the project. Click *select options* to configure the **Conflict Builder** settings.

Edit Drive to Watch

Select this link to launch the ***Edit Drives to Watch*** dialog. Select any drives that the application may use during its installation. The selected drives will be monitored during the installation of the application.

Edit Exclusions List

The exclusions list contains files and registry keys which will be ignored during the file system monitoring. Select this link if you wish to edit the exclusion lists to include specific files, registry keys or folders.


Start the Repackaging Process

Click here to start the repackaging process.

MODIFY AN EXISTING MSI

There wizard allow the creation of a transform that is used to modify an existing MSI file.

SELECT BASE MSI

Provide the full path and file name to the MSI file that the transform will be based on. Press  to browse to the file.

HOW TO CREATE TRANSFORM

Create a transform directly based on an MSI

Select this option to make specific changes to the MSI. This option will open the base MSI and allow you to make changes to it. When the project is saved, the changes will be saved to the transform file specified in the SAVE AS section of the dialog. The base MSI will not change.

Example: Typical scenarios might include the need to add some registry keys that are required by your organization or to set a property required by a particular department.

Create a response transform (used for silent installations)


Select this option to capture the selections and configurations that are made within the user interface part of the base MSI install.

For example, chances are that an application from a vendor in the form of an MSI might need the default install directory changed, default features to install and how (advertised, run from network, install locally) and other options such as not to run the application after installation. The configurations you make depends on the options are given during the user interface phase of the base MSI install.

Create a transform from the difference of 2 MSIs


This option is provided in order to create a transform based on the differences between 2 MSI files. Often it is easier to maintain the base MSI and add changes to a copy of this. Before release or distribution you can create the transform from the 2 MSIs you have.

This option can also be used as a tool to find the differences between 2 MSIs.

Specify the name of the second MSI file. Press  to browse for the second MSI file.

When the GO button is selected a transform will be created. If the transform is created successfully, it will be opened along with the base MSI so further modifications can be made if necessary.

SAVE AS

Specify the name of the newly created MST transform file. Press  to browse to a path and specify a new file name. The file must have an MST file extension. This MST file will be created at the end of the repackaging process.

GO

Click GO to begin the process of creating the transform.

TEST THE INSTALLATION

Often MSI files are embedded within a setup EXE for easy deployment. When you run the executable, it is not always obvious whether the installation is Windows Installer based.

This wizard will detect whether the installation is Windows Installer based and will lead you to the appropriate next step.

Select the setup executable in the entry box below and press GO to launch it.

Wait for any files to be extracted and then step through a few screens of the installation as you normally would, until you reach the license agreement.

Once you are sure all files have been extracted press the Test button. A dialog will let you know whether the install is MSI based. If it is you are given the opportunity to copy the MSI and dependent files to a network folder.

DO NOT INSTALL THE APPLICATION.

DO NOT CANCEL THE INSTALLATION UNTIL AFTER YOU HAVE TESTED AND COPIED THE FILES.

CREATE A WRAPPER INSTALLATION

This process is used to wrap installations which cannot be repackaged. Installations should not be repackaged are if they are installing settings which differ on different machines, or if they are updating windows protected system files. Examples might include windows updates and security patches. Changes should not be captured but rather the original installation or patch should be wrapped in a Windows Installer MSI file for deployment.

SAVE AS

Provide the full path and file name of the MSI file you wish to create. Press to browse to a path and specify a new file name. The file must have an MSI file extension. This MSI file will be created at the end of the repackaging process.

SELECT FILE TO WRAP

Provide the full path and file name of the file that the MSI wrapper will be created for. Press to browse to the file.

COMMAND LINE

Some installs have command line parameters that must be used to run or customize the installation. Enter any command line parameters that will be used when running the executable installation. These command line parameters will be added to the newly created MSI as options.

UNINSTALL

Do not run the executable on uninstall

Select this option if no uninstall process is necessary for this application.

Run the executable on install

Select this option if the wrapper should include an uninstall process.

GO

Click GO to begin the process of creating the new MSI file. Once created it will be opened in order to add any other settings that may be needed in the project such as files and registry entries.

CREATE NEW SETUP INSTALLATION

This selection will create a new MSI project. When starting a new project you can begin with an empty MSI project or with a *template*. A template is a base MSI file that contains standard entries such as dialogs, standard properties and error messages. Templates are then used as the starting point when creating a new MSI. New entries are added to the template to make the finished MSI.

MSI Studio comes with two basic templates -- Repackage Template and Windows Template. If neither template suits your needs, they may be customized or a new template created.

To create a new template, choose to create a new MSI within MSI Studio and make your template settings. Save this new file to the templates folder. The templates folder, by default, is created on the local machine in the \Documents and Settings\User\Application Data\Scriptlogic\IDS\templates\ folder. This folder can be changed by selecting the Tools > Options menu items. On the options form, select the Templates tab.

Select a Template

To begin a new MSI project with a pre-defined template, click the template link in the list.

Open the new project directly

To begin a new MSI project from scratch (with no template), click *Open the new project* directly below the Template list.

OPEN EXISTING PROJECT

This selection will open an existing Windows Installer project (MSI), Windows Installer Transform (MST), or other vendor project databases such as WSE's or ISM's in the project editor. Select a project from the Recent Projects list, or browse to an existing project by clicking the *Browse for an existing project* link.

Inside the MSI

TABLES

The Tables tab defines all tables that make up the detailed information about the installer package. These tables contain information that has been updated throughout the project and are used to compile the final Installer package.

It should not be necessary to modify these files, however, should the need arise to do so, make sure you fully understand the tables and columns that are to be modified. A full description of all tables and columns can be found in the Microsoft Platform SDK Windows Installer help file.

IQ VIEWS

Overview

The IQ Views tab is divided into two sections. The left-hand pane is comprised of five project sections. These sections include

- Project Details
- Project Organization
- Files and Registry
- Shortcuts and Other Items
- Dialogs and Actions

Project Details

Contains properties about the package. This section includes Product Details, Add/Remove Programs, Upgrades and Application Requirements.

Project Organization

Displays the package components and features. View individual entities within a component/feature (i.e. files in a component) by expanding out the components/features. Entities can be moved between components/features by a drag and drop action. To add or delete components/features right-click on the specific component/feature item.

Files And Registry

Contains panels for Files, Registry entries, Merge Modules, Permissions and Path Variables.

Shortcuts and Other Items

Contains the projects Shortcuts, INI Files, ODBC settings, Environment Variables, Service Management, File Extensions and COM Advertising. Expand each top level node to view the details of each item. New Items can be added or deleted by right-clicking on the item.

Dialogs and Actions

This section contains the Dialogs, Event Sequencing and Custom Actions defined in the package.

Project Details

Product Details

The Product Details dialog is split into 3 sections -- Product Details, General Information and Installation Properties.

The Product Details section contains required properties of the MSI.

Name

Relates to the ProductName property and should reflect the name of the package. Generally the application vendor's name should not be included. For example, 'Acrobat Reader' - not 'Adobe Acrobat Reader'.

Version

The version number should be in the form major.minor.build and should only contain numerals and '.'s.

Manufacturer

The name of the software vendor. e.g. ScriptLogic Corporation

Default Directory

When creating an installation program that is to be installed manually you need to set a default directory only if the user gets the option to select where to install the product. When repackaging for deployment around a networked environment it is not necessary to set this and we recommend you leave it blank.

Product Code

Each product release has a unique identifier. This is set automatically so does not need to be set, however, this can be done by clicking the **Create** button.

General Information

The General Information section contains information which is added to the Summary Information Screen and is displayed when the mouse hovers over the file in Windows Explorer.

Title

Briefly describes the type of installer package. Commonly this is set to the same as the Name in the Product Details section.

Subject

This field provides further details about the package.

Author

The person, team or company who is creating the package.

Keywords

These words are used when doing keyword searches within Windows Explorer.

Comments

Enter any further comments about the package.

Package Code

Enter a unique package identifier (GUID). Click Create to create a new identifier for the package.

Add or Remove Programs

The Add or Remove Programs dialog contains settings that affect what will be displayed in the Add/Remove Programs control panel for the package.

In most locked-down environments users won't have access to the control panel so it's not always important to modify these properties. We recommend that you set them in the template and then leave the information the same for all packages.

Add or Remove Programs

Display in Add or Remove Programs

Prevents the application from displaying in the Add or Remove programs list.

Hide Modify Button

Select this box to hide the Modify button for the application in the Add or Remove Programs list.

Hide Repair Button

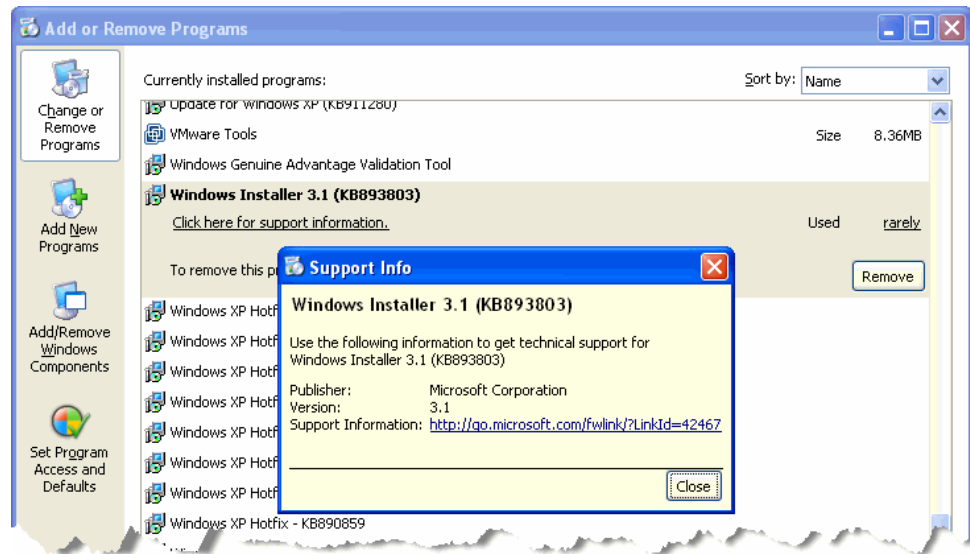
Select this box to hide the Repair button for the application in the Add or Remove Programs list.

Hide Remove Button

Select this box to hide the Remove button for the product in the Add or Remove Programs list.

Support Information

In the Add or Remove programs Control Panel applet, each application has the ability to show support information. The entries in the IDS Support Information section allow the specification of the values that will display in the Support Information dialog.



Contact

The contact information for support with the application.

Phone Number

The phone number to use for support with the application.

Support URL

The URL to use for support with the application.

Updates URL

The URL to use for updates of the application.

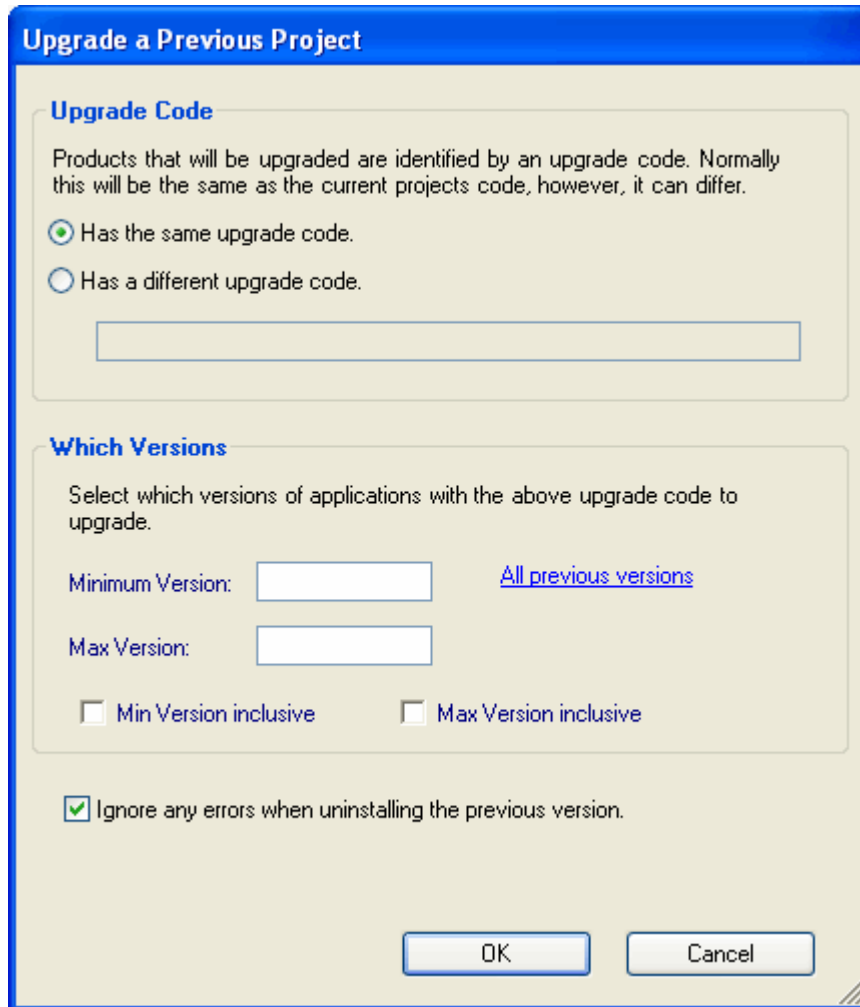
Comments

Additional support information.

Upgrades

MSI files can be used to update an existing installed application. The Upgrades dialog lists the projects that will be upgraded by this project. Click **Add** to add a project to the Upgrade list. Click **Remove** to delete a project to the Upgrade list.

Adding a Project to the Upgrade List



The screenshot shows a dialog box titled "Upgrade a Previous Project" with a blue header. It contains two main sections: "Upgrade Code" and "Which Versions".

Upgrade Code
Products that will be upgraded are identified by an upgrade code. Normally this will be the same as the current projects code, however, it can differ.

Has the same upgrade code.
 Has a different upgrade code.

Below the radio buttons is an empty text input field.

Which Versions
Select which versions of applications with the above upgrade code to upgrade.

Minimum Version: [All previous versions](#)
Max Version:

Min Version inclusive Max Version inclusive

Ignore any errors when uninstalling the previous version.

At the bottom right are "OK" and "Cancel" buttons.

Select *Has the same upgrade code* if the upgrade code is the same as the project code. Most often, this will be the case. If the codes are different, select *Has a different upgrade code* and specify the new upgrade code.

Versions

Minimum Version

Specify the minimum application version to include in the upgrade.

Max Version

Specify the maximum application version to include in the upgrade.

Min Version inclusive

Select this box to include the minimum application version in the upgrade version check.

Max Version inclusive

Select this box to include the maximum application version in the upgrade version check.

Ignore any errors when uninstalling the previous version

Select this box to ignore application errors when uninstalling previous versions of the application.

Application Requirements

The Application Requirements dialog specifies requirements of the target machine before an application can be deployed to it. These requirements include Operating System, Internet Explorer and .NET Framework minimum versions. System requirements can also include specific applications that are required before the project is installed.

Operating Systems

Minimum Windows 9x Version

Select the minimum required 9x operating system required on the target machine before the application is installed. Select from *No check made*, *Windows 95*, *Windows 98* or *Windows ME* from the drop list.

Minimum Windows NT Version

Select the minimum required 9x operating system required on the target machine before the application is installed. Select from *No check made*, *Windows NT 4.0*, *Windows 2000*, *Windows XP*, *Windows 2003 Server* or *Windows Vista* from the drop list.

Minimum IE Version

Select the minimum required 9x operating system required on the target machine before the application is installed. Select from *Do not check*, *IE Version 4.0*, *IE Version 5.0*, *IE Version 6.0*, *IE Version 6.0 (SP2)* or *IE Version 7.0* from the drop list.

Minimum .NET Framework

Select the minimum required 9x operating system required on the target machine before the application is installed. Select from *Do not check*, *.NET Framework Version 1.0 3705*, *.NET Framework Version 1.1.4322* or *.NET Framework Version 2.0 50727* from the drop list.

Application Search

New Application Search

Search For a File or Folder

File Name: ..

Folder Name:

Folder Search Depth:

Version Min: Version Max:

Date Min: Date Max:

Size Min: Size Max:

Search For a Registry Entry

Registry Root: ▾

Registry Key:

Value Name:

Property Name:

Create a Launch Condition based on the search criteria.


Message:

OK Cancel

Search for a File or Folder

Select Search for a File or Folder to create a requirement of a certain file or folder within an application. File or directory searches can be used to determine whether a user has already installed a version of an application.

File Name

Enter the path and file name of the required file. Click  to browse for the specific file.

Folder Name

Enter the required folder name.

Folder Search Depth

Enter the depth below the specified folder that the installer will search for the file or folder.

Version Min

The minimum version of the required file. If a version is specified, then the file must have a version that is at least equal to the minimum version specified.

Version Max

The maximum version of the required file. If a version is specified, then the file must have a version that is at most equal to the maximum version specified.

Date Min

The minimum modification date and time of the required file. If this field is specified, then the file must have a modification date and time that is at least equal to the specified minimum date.

Date Max

The maximum modification date and time of the required file. If this field is specified, then the file must have a modification date and time that is at most equal to the specified maximum date.

Size Min

The minimum size of the required file. If a size is specified, then the file must have a size that is at least equal to the minimum size.

Size Max

The maximum size of the required file. If a size is specified, then the file must have a size that is at most equal to the maximum size.

Search for a Registry Entry

Registry Root

Select *HKEY_CLASSES_ROOT*,
HKEY_CURRENT_USER,
HKEY_LOCAL_MACHINE, *HKEY_USERS* or
HKEY_USER_SELECTTABLE from the drop list.

Registry Key

Specify the registry key to search for.

Value Name

Specify the registry key value to search for.

Property Name

Enter a Property Name to be set when the file or registry entry is found. This property can then be used in a launch condition. For example, a feature might have a condition to only install if the property is set to 1 (item has been found).

Create a Launch Condition based on the search criteria

A Launch Condition is a condition that is checked upon start up. If the condition is not met the message is displayed and the installation is cancelled. If a launch condition is selected, the file or registry entry will be searched for on startup and if not found the installation will not be run.

Project Organization

Features

A feature is a part of the application's functionality that is able to be installed independently of the entire application. Examples of features could be a spell-checker, set of clip art or dictionary.

Usually, when packaging for a networked environment there will only be one feature (the application) and it will always be installed.

When installing an application, features are presented to the user in the form of dialogs which features are available, selection for installation and various prompts for information on how to install them.

Name

This is the key name of the feature and should be unique.

Title

The name of the feature as it will be displayed in the dialog during the user interface sequence of the installation. When packaging for a networked environment the user interface is not displayed therefore this is not important.

Parent Feature

If the feature requires some other feature to be installed before this feature can be installed (parent), specify the parent feature for this child.

Description

A description of the feature.

Display

Specify how the feature is initially displayed for selection in the User Interface. Select from *Visible and Expanded*, *Visible and Collapsed* or *Hidden*.

Level

The initial level of the feature and how it will initially be installed. Select from *Normal*, *Never install this feature*, *Always install this feature* or *Custom*.


Attributes

Further options on how to install the feature. Select from *Favor Local*, *Favor Source* and *Favor Parent*. *Favor Local* means that if the feature is selected, it will be installed locally. *Favor Source* will be installed to run from the source CD-ROM or server. *Favor Parent* will force the feature to be installed and run the same as its parent feature.

Advertising

Advertising provides the ability to make certain features available within an application even though the feature has not yet been installed. Once the feature is activated or selected, the feature automatically installs the necessary components. Select from *Favor Advertising* or *Disallow Advertising*.

Directory

Features can be installed into different folders. Specify the default directory for this feature here. Click  to browse to the folder.


Components

Windows Installer installs and removes applications in pieces. These pieces are the application components. Components are made up of a set of resources such as files, registry entries, shortcuts or anything else that is to be installed.

Name

This is the name of the component.

Directory

The directory should be specified as the folder where the files will be installed to. Click  to browse to the folder within the project.

GUID

The GUID is a unique identifier for the component and is set when a component is created. Click **New** to create a new GUID.

Condition

A component can conditionally be installed. If the condition specified is Null or TRUE the component is enabled and installed. If the condition evaluates to FALSE, the component is not installed.

Key Path Type

Select the type of key path for the component. Select *File Key Path*, *Registry Key Path*, *ODBC Data Source Key Path* or *Directory of the Component* from the drop list.

Key Path

Having selected the key path type above, select the key path. If you have selected a key path type of File Key Path, then all the files in the component are listed in this field.

TIPS

Component Key Paths are one of the most important things to consider when you're packaging an MSI. Whenever an application is run via an advertised entry point (i.e. an advertised shortcut) a check is made of all the components in the feature that the entry point points to.

The check consists of detecting whether the key path for that component is present. If the key path is not present, then it is repaired. Therefore, it is vital that the key paths are correct. All vital files should be in their own components and be key paths (.exe, .dll, .ocx files). If this is done then they will always be present when the application is run, even if deleted

accidentally.

It is also important not to have a file or a registry key that might be deleted, either by the user or an application, as a key path. A common error that occurs in MSI packages is that every time an application is run it repairs itself. This can be caused by a component in the package having a key path which is deleted by the application. For example, having an ini file which is deleted when the application is closed down as a key path will cause the package to repair, and replace the missing file, every time it is launched.

A useful tip to ensure that user-specific information is present is to have a 'Current User' component that has a HKEY_CURRENT_USER registry key path. When a new user launches the application a check will be made for the registry key, detect it if not there and then repair the application, ensuring that the new user has all the necessary settings.

Attributes

Always increment dll count

Select if the key file of the component is a shared dll.

Leave installed on uninstall

Select if the component is permanent and you never wish to remove it.

Check condition on reinstall

When reinstalled, the condition of the component is reevaluated.

Never overwrite if key path exists

Select if the component should never be overwritten.

Files and Registry

Files

The files view allows files to be added to, removed from, and moved within the package being edited. It also allows for folders to be created, removed and renamed.

The File panel is split into two sections

- System View -- displays files and folders on the local machine
- Project View -- displays files and folders within the package

Files and folders can easily be added from the System View to the Project View.

Project View


Windows Installer contains a number of special system folder properties which should be used as much as possible. Common examples are the Program Files folder and the System Files Folder. These system folders are displayed under the 'Special Folders' folder in the Project View. This 'Special Folder' actually represents the TARGETDIR property, so if you want to create a new folder under the TARGETDIR then create it here.

New folders can be added to the project in a few ways. Right-click on the parent folder in the Project View and select 'New Folder' from the context menu or select the parent directory in

the Project View and then click . Enter the folder name in the *Create New Folder* dialog.

Add a New Root Drive to the Project

If you do not wish to use a system folder then create a new hard coded root drive property, from which you can create subfolders. Right-click anywhere in the System or Project folder view and select 'Add New Root Drive' from the context menu or

click . Enter the drive letter in the form of x:\, where x is the root drive.


Delete a Folder

Right click on the folder you wish to delete in the Project View and select 'Delete Folder' from the context menu or select the

folder and then click .

Add a Folder from Local System to Project

To add a folder (including all sub-folders and files within it) from the local system to the project, select the Project View folder where it is to be added. Next select the folder to be added in the System View and

- from the context menu of the folder in the System View select 'Add Folder' OR
- Click  OR
- Drag the folder from the System View to the folder where it is to be added in the Project View.


This will add all sub folders and files within the selected folder as well.

Rename a Folder

Select the folder in the Project View and press F2 or right-click on the folder and select Rename Folder from the context menu. The folder name will then be editable allowing its renaming.


Add an Individual File from the Local System to Project

To add a file from the local system to the project, select the folder in the Project View where the file is to be added. Next, in the top System View, browse to the appropriate folder and select the file in the top right panel. Add files by

- double clicking the file or
- dragging the file to the bottom right panel or the appropriate folder in the Project View or
- clicking  or
- selecting *Add File* from the context menu for the System View file.

Delete a File

Select the file in the Project View and either

- press the delete key or
- from the context menu select *Delete File* or
- click  in the File View

Display or Edit File Details

To display or edit the file details double click the file in the Project View. Select *Details* from the context menu for the file.

File Details

The File Details dialog contains information about the properties of a file within the project and a path to the source file.

Filename

The file name used for installation.

Component

Select the component that the file belongs to from the drop list.

Source

The path to the source file. This is used when building the project.

File Attributes

Read-only, Hidden, System and *Vital* are attributes that the file will be installed with. Select the necessary attributes.

Valid-Checksum

Select this box if the file has a valid checksum.

Patch Added

Select this box if the install is a patch.

Non-compressed

Select to override the package's main compressed property. This attribute should be left unchecked unless absolutely necessary.

Compressed

Select to override the package's main compressed property. This attribute should be left unchecked unless absolutely necessary.

Registry

The Registry View allows the adding, editing and deleting of registry keys, names and values in the currently opened package. The panel is split into two views, the top view which is the current registry view of the local system, and the bottom view which is the current project. Entries from the local system view can be added to the project view.

Adding a Registry Key

A registry key can be added by adding an existing key, adding an existing key with its sub keys or creating a new key to add to the project.

To add an existing key, with or without sub keys, select the key in the top System View. It can be added either using the buttons (*Add Key* or *With Sub Keys*) on the panel, or by right-clicking and selecting the option on the context menu. All values belonging to the key will also be added.

To create a new key, select the parent key in the bottom Project View and then right-click to bring up the context menu and select *Add Key*.

To Add a Registry Value

Registry values can either be added or created. To add an existing value, browse to it in the top System View and then add it by double-clicking, select the option in the context menu, or use the button on the panel. The key for the value will also be added.

To create a new value, in the right hand panel of the Project Value, use the context menu to select *New>String/DWORD/Binary* value.

Remove/Editing Keys and Values

There is an option to delete and edit key names in the context menu, or by pressing F2 or the delete key.

View or Edit Registry Value Details

Select the value in the bottom left Project View panel and either double click or bring up the context menu and select *Details*. This brings up the Registry Entry Details dialog box.

Remove a Registry Key on Install

Add the registry key to the project and then select it and bring up the context menu. There is an option which should be selected (checked) called *Remove all subkeys and values on Install*.

Remove Value Entry during Install

To remove a value during the install, bring up the context menu for the value and check the item *Delete Name on Install*.

Importing and Exporting Registry Keys and Values

There are buttons and options in the context menus to import from registry files or export to registry files. The exported registry files are in the same format as those exported from regedit and vice versa.

Merge Modules

Merge modules are used to deliver shared code, files, resources, registry entries, and setup logic to applications as a single compound file. A single merge module can be used in multiple projects.

The Merge Module panel allows you to add Merge Modules to and from the current project. It is divided into two sections. The top half lists the modules that are present on the local system. The bottom half is what is contained in the project.

To modify the search location of modules on your system go to Tools>Options>Merge Modules.

The *Feature* drop list allows you to select the feature you are working with. It will display all the modules in that feature.

To Add a Module

Modules can be added by double clicking them in the top panel, by selecting and using the context menu option or by clicking the button provided.

To Remove a Module

To remove a module select it in the bottom panel and then press the Remove button, or use the option in the context menu.

Refresh Button

The refresh button will refresh the list of modules in the search locations. Therefore if you modify the module search locations then press this button.

Permissions

The Permissions panel allows you to create files, folders and registry keys and set specific permissions on them when installing a package.

File & Folder Permissions

Folder List

Select a file or folder from the File & Folder list. The selected File or Folder will be the recipient of the specified permissions. Right click in the File and Folder list for Folder options.

New Folder (INS)

Right click and select *New Folder* or press *INS* to create a new folder.

Add New Root Drive

Right click and select *Add New Root Drive* to create a new root folder. The new folder should be in the form of *c:* and must contain a trailing backslash.

Delete Folder (DEL)

Right click and select *Delete Folder* or press *DEL* to delete a folder.

Rename (F2)

Right click and select *Rename* or press F2 to rename a folder. Clicking on the folder name will also allow a folder rename.

Hide Empty Folders

Right click and select *Hide Empty Folders* to hide all folders that do not contain files.

Expand All

Right click and select *Expand All* to expand all folders.

Collapse

Right click and select *Collapse* to contract all folders.

Create as Empty Folder

A Folder that exists in the MSI project is not created during installation, by default, unless there are files in the folder. Set *Create as Empty Folder* to force the installation to create the folder even if there are no files that belong in it.

Users\Domain List

Add

Click **Add** to add a new User\Domain to which the permissions will apply.

Delete

Click **Delete** to delete a User\Domain from the list.

Details

Click **Details** to update the User and Domain details.

Permissions List

The following permissions can be set for a file or folder.

- Generic All
- Generic Execute
- Generic Write
- Generic Read
- Execute File
- Read Data
- Read Attributes
- Read Extended Attributes
- Write Data
- Append Data
- Write Attributes
- Write Extended Attributes
- Delete Files
- Delete
- Read Permissions
- Change Permissions
- Take Ownership

Select None

Click **Select None** to clear all permissions.

Select All

Click **Select All** to set all permissions.

Registry Permissions

Registry List

Select a Registry key from the Registry list. The selected Registry Key will be the recipient of the specified permissions.

Users\Domain List

Add

Click **Add** to add a new User\Domain to which the permissions will apply.

Delete

Click **Delete** to delete a User\Domain from the list.

Details

Click **Details** to update the User and Domain details.

Permissions List

The following permissions can be set for a file or folder.

- Query Value
- Set Value
- Create SubKey
- Enumerate SubKey
- Notify
- Create Link
- Delete
- Write DAC
- Write Owner
- Read Owner

Select None

Click **Select None** to clear all permissions.

Select All

Click **Select All** to set all permissions.

Path Variables

Path variables are variables that are used with a project to define commonly used path values. Once defined, the variable can be used throughout the project. The great thing about these variables is that if the value needs to change, it only needs to be changed in the Path Variables panel **once**. The new value will be used throughout the project, wherever the variable is used.

Lets say that your projects' files are stored in E:\MyApplicationSource\Files and E:\MyApplicationSource\Images. If no Path Variables were to be used, each path would be hard coded into the project. Hardcoding the path can result in two problems. First, if it is misspelled, the application will not be written to the MSI file correctly. Second, if this path changes, it will need to be manually changed for every file from these folders that is included in the project.

Instead of hard coding the path value, use a Path Variable to define it once. Create a Path Variable. To continue with the example used above, we can create a Path Variable called SourceLocation and set it to E:\MyApplicationSource. Now when files are included in the project, the Path Variable is used instead of the full hardcoded path. If the source files move to a new location at a later time, just the variable needs to be updated.

Shortcuts and Other Items

Shortcuts


There are two types of shortcuts, advertised and command line. Advertised shortcuts create windows installer link files that point to a component within a feature of an MSI file. When it is run the windows installer service performs a series of consistency checks. A command line shortcut is a standard shortcut which points directly to a file.

Click **Add Advertised** to add an advertised shortcut. Click **Add Command Line** to add a command line shortcut. To delete a shortcut, highlight it in the list and click **Delete**. To view a shortcut's details, highlight the shortcut in the list and click **Details**.

Name

Enter the name of the shortcut. This should be a unique name.

Target File

For advertised shortcuts this should contain the key to a file which is to be launched by the shortcut. Click  to browse to and select a file within the package.

For command line shortcuts, enter the full path to the file which will reside on the target system.

Dest. Directory

This is the folder where the shortcut will be created. Normally this will be a subfolder of the Program_Files_Folder folder (which appears in the start menu), or the Desktop Folder.


Arguments

Enter any command line arguments that you wish to run the target file with.

Description

Enter the description which will be displayed when the mouse hovers over the shortcut.

Working Directory

If you wish to specify a working directory for the target file to start in, click  to browse to a folder within the package.

Show Window

Select whether to start the window maximized, minimized, or normally.

Component

Select the component to add the shortcut to.

Icon

Click **Browse** to locate the file that contains the icon, or to the icon file itself.

Ini File

INI file entries should be added through this interface rather than installing a file that has an extension of .INI unless it is certain that the INI file will not be used by any another application.

File Name:

Enter the name of the INI file to be updated.

Directory:

Browse to the folder in the package where the INI file will be installed to.

Component:

(Only on the edit dialog) Select the component that the INI file belongs to from the drop list.

Feature:

(Only when adding a new INI file) Select a feature to add the new INI file to.

Action:

Select that action that you would like to take when adding the INI file. Select from *Create or update the entries*, *Create if does not exist* or *Create and append (comma separated)*.

Create or update the entries

Creates or updates a INI entry.

Create if does not exist

Creates a INI entry only if the entry does not already exist

Create and append

Creates a new entry or appends a new comma-separated value to an existing entry.

ODBC

Adding an ODBC DataSource to Project

When creating an ODBC Datasource, best practice says It is best to import the ODBC Datasource entries from the local computer. That way, entries will always be correct.

Data Source:

Enter the name of the data source. Click **Import** to import an existing ODBC Datasource from the local machine.

Driver:

Enter a description of the driver.

Registration:

Select to register the data source per user or per machine.

Driver Attributes:

Add the driver attributes that are required.

Add ODBC Driver to Project

Driver Name

Enter the name of the driver or click **Import** to import all of the entries from a driver on the local machine.

Driver .DLL

Click **Browse** to locate the file within the package which is the driver file. If the driver was imported then the file will automatically be added to the project.

Setup .DLL


Click Browse to point to a file which acts as the setup file for the driver. If the driver was imported then the file will automatically be added to the project.

Source Attributes


Enter any required driver attributes.

Add ODBC Translator to Project


Description

Enter the description for the translator file or click  to locate a translator file on the local machine.

Translator File

Click  to point to the translator within the package. If the translator was imported from the local machine then this file will automatically be added to the package.

Setup File

Click  to locate the file within the package that acts as the setup file for the translator. If the translator was imported from the local machine then this file will automatically be added to the package.

Environment Variables

The Environment pane allows the application to create environment variables when it is installed.

Name

Enter the name of the environment variable to be created/edited.

Value

Enter the new value or the value to be appended to the environment variable.

Action On Install

Select from *Create or Update*, *Create if it does not exist* or *Remove* to create an environment variable during install.

Action on Uninstall

Select to *Leave* or *Remove* the environment variable during an uninstall.

Replacement

Select how the value is to be added. Select from *Replace current value*, *Append at end of current value* or *Insert at beginning of current value*.

Feature

Select the feature to add the environment variable to from the drop down list.


Windows NT/2000/XP System Environment Variable

Select this check box if the environment variable is a system environment variable. If it is not checked it will be installed as a user environment variable.

Service Management

Add New Service to Project

Service File

Click  to browse and select the file within your package that is the service executable file.

Service Name

Enter the service name.

Display Name

Enter the name to be display in the service control panel.

Description

Enter a description to be displayed in the service control panel.

Arguments

Enter any arguments that the service executable should be run with.

Login Name

The name that the service is logged on as.

Password

Enter the password for the account.

Load Order

If part of a load ordering group then enter the group name here.

Dependencies

If the service has any dependencies then enter them here.

Error Control

Select from *Ignore Error*, *Normal Error* or *Critical Error* from the drop list. Select *Ignore Errors* to log the error and continue with the startup operation. Select *Normal error* to log the error, display a message box and continue with the startup operation. Select *Critical Errors* to log the error if it is possible and restart the system with the last known good configuration. If the last known good configuration is being started, the startup operation will fail.

Process Type

Select the process type.

- **Runs in its own process** -- Select if the service is a Microsoft Win32 service that runs its own process.
- **Shares a process with others** -- Select if the service is a Win32 service that shares a process.

Start Type

Select whether the service should start Automatically or Manually.

Service interacts with desktop

Select this box if the service is a Win32 service that interacts with the desktop.

Add New Service Control to Project

Service install controls are used to start and stop services during the installation. For example, following the installation of a service you might want to start it, or before uninstalling a service you might need to stop it.

Service Name

Select a service from the drop list that already exists in the package.

Arguments

Enter a list of arguments for starting the service separated by the ~ character.

Feature

Select the feature to add the service to.

On Install

Select whether you wish to *Start*, *Stop* or *Delete* the service on install.

On Uninstall


Select whether you wish to *Start*, *Stop*, or *Delete* the service on uninstall.

Wait for action to complete before continuing

Select this box to pause the action and wait for the service Start, Stop or Delete to finish before continuing.

File Extensions

Executable File

Click  to select a file already within the package that is to be associated with the file extension.

Extension

Enter a name for the extension. For example, 'doc'

ProgID

If the file extension is associated with a ProgID enter it. If it is already in the package, select it from the drop list.

Description

Enter a description of the ProgID.

Change Icon

Click Change Icon to select an Icon from the file associated to the ProgID.

COM Advertising

Add an application ID to the Project

App Id

The AppId value that will be written under the CLSID and creates the AppId GUID key under HKCR\AppId.

Remote Server Name

This column contains the value of "RemoteServerName"=<xxxx> that will be written under HKCR\AppID\{AppID}\ .

Local Service

This column contains the value of LocalService that will be written under HKCR\AppID\{ <appid> } "LocalService"= <xxx> .

Service Parameters

This column contains the value of ServiceParameters that will be written under HKCR\AppID\{appid>} "ServiceParameters".

DLL Surrogate

This column contains the value of DIISurrogate that will be written under HKCR\AppId\{ <appid> } "DIISurrogate"= <xxx> .

Add a Class ID to the Project

Class ID

Enter the Class identifier of a COM server.

Context

The server context for this server.

App ID

Application ID containing DCOM information for the associated application.

Prog ID

Enter the Program ID

Description

Description associated with the Class ID.

File Type Mask

Contains information for the HKCR (this CLSID) key.

DefInprocHandler

Default inproc handler.

Component

Specify the feature that provides the COM server.

Argument

This column is optional only when the Context column is set to the LocalServer or LocalServer32 server context. The text is registered as the argument against the OLE server and is used by OLE for invoking the server.

Attributes

If msidbClassAttributesRelativePath is set in this column, the bare file name can be used for COM servers.

Icon

The file providing the icon associated with this CLSID.

Add a Program ID to the Project

Prog ID

Enter the Program ID.

Parent ID

Enter the Program ID for version independent programs.

Description

Enter a description of the program.

Class

Enter the Class Identifier.

Icon

Select a program associated icon.

Add a TypeLib to the Project

TypeLib GUID

The GUID that identifies the library.

Language

The language of the type library.

Component

The component whose key file is the type library being registered.

Version

This is the version of the library.

Description

The description of the library.

Directory

This column identifies the Help path for the type library.

Cost

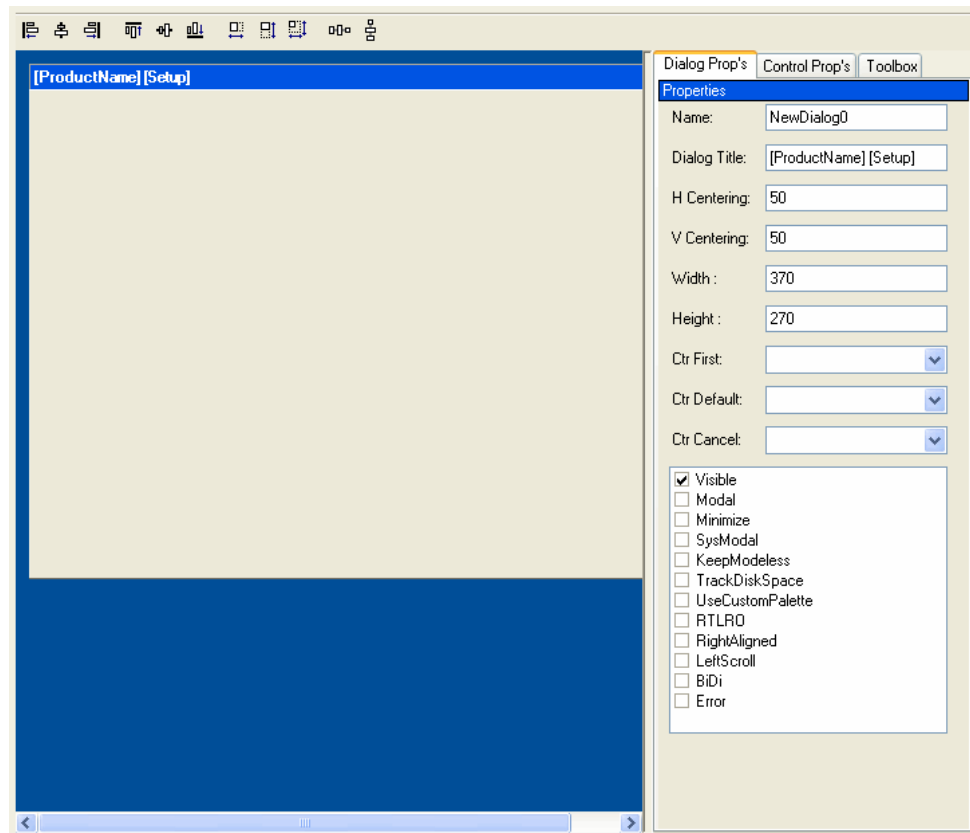
The cost associated with the registration of the type library in bytes.

Dialogs and Actions












Dialogs

The Dialog Editor is used to create Dialogs within a package. These dialogs are the user interface part of the installation package.

The Dialog Editor



Toolbar

- | | | | |
|---|--------------|---|-------------------------|
|  | Align Left |  | Horizontal Size |
|  | Align Center |  | Vertical Size |
|  | Align Right |  | Horiz/Vert Size |
|  | Align Top |  | Horizontal Distribution |
|  | Align Middle |  | Vertical Distribution |
|  | Align Bottom | | |

Dialog Prop's

Name

The Name of the dialog box.

Dialog Title

The Title to be displayed in the title bar of the dialog box.

H Centering

The Horizontal position of the dialog box.

V Centering

The Vertical position of the dialog box.

Width

The Width of the dialog box.

Height

The Height of the dialog box.

Ctrl First

Select the control that gets the focus when the dialog box is opened.

Ctrl Default

Select the default control that gains focus when the dialog box is opened. Normally this is a PushButton control. Pressing the Return key is equivalent to clicking on the default control.

Ctrl Cancel

Select the control that cancels the installation. Hitting the ESC key or clicking the Close button is equivalent to clicking on the cancel control.

Dialog Attributes

Select one or more of the attributes to be used for the dialog.

Visible

Select this box to open the dialog and display it to the user, otherwise it will be hidden from the user.

Modal

Select this box to disallow other dialogs of the same application to be put on top of it; otherwise the dialog is modeless and other dialogs of the same application may be opened on top of it

Minimize

Select this box to provide the ability to minimize the dialog box.

SysModal

Select this box to make the dialog a System Modal dialog box. The dialog box will stop all other applications and no other applications can take the focus.

KeepModeless

Select this box to allow other dialogs stay alive when this dialog box is created.

TrackDiskSpace

This attribute can be used if there is a control on the dialog indicating available disk space. If the user switches to another application, adds or removes files, or otherwise modifies available disk space, you can quickly implement the change using this style.

UseCustomPalette

Use of this attribute displays pictures on the dialog box with the custom palette (one per dialog received from the first control created). If the attribute is not set, the pictures are rendered using a default palette.

RTLRO

Use of this attribute will display the text in the dialog box in right-to-left-reading order.

RightAligned

Use of this attribute will display the text in the dialog box on the right side of the dialog box.

LeftScroll

Use this attribute to position the scroll bar on the left side of the dialog box.

BiDi

This attribute is a combination of the right to left reading order (RTLRO), the RightAligned and the LeftScroll dialog attributes.

Error

Use of this attribute dictates the dialog is an error dialog. The error dialog must have a static text control named ErrorText. This control receives the text of the error message. The dialog should also have the seven push buttons corresponding to the possible return values. The error message determines which of these buttons are actually displayed. The displayed buttons are rearranged so they are evenly distributed on the dialog. This rearrangement changes the X coordinate of the buttons, but not the other three coordinates. Therefore it is advisable that no other control should be authored in the same horizontal region of the dialog as the buttons. If the error message specifies no button, the OK button is displayed. The Default button, First active control and Cancel button values are ignored for an error dialog. The Default button defined in the error message will be assigned to all three values.

Control Prop's

X

The horizontal coordinate of the upper-left hand corner of the control.

Y

The vertical coordinate of the upper-left hand corner of the control.

Height

The height of the control.

Width

The width of the control.

Text

Specify text to be displayed on the dialog box.

Help

Specify text strings that are to be used with the Help button. The specified string is divided into two parts by a separator character (|). The first part of the string is used as ToolTip text. This text is used by screen readers for controls that contain a picture. The second part of the string is reserved for future use. The separator character is required even if only one of the two kinds of text is present.

Font

Select the font to be used for the control.

Property

Select a defined property to be linked to the control from the drop list.

Graphic

For controls which allow images, select the image to be displayed.

Control Events

Click **Add** to add a control event to the list to have an event specify an action to be taken by the installer. Click **Remove** to remove a control event from the list.

Conditions

Click **Add** to add a condition to the list to Enable, Disable, Show or Hide the control based on a specified condition. Click **Remove** to delete a previously defined condition from the list.

Toolbox

The following controls can be added to a dialog within the package. Clicking on a control will cause it to be automatically added to the dialog.

- **Dialog**
- **Billboard**
- **Bitmap**
- **CheckBox**
- **ComboBox**
- **DirectoryCombo**
- **DirectoryList**
- **Edit**
- **GroupBox**
- **Icon**
- **Line**
- **ListBox**
- **ListView**
- **MaskedEdit**
- **PathEdit**
- **Progress Bar**
- **PushButton**
- **Radio Button Group**
- **Scrollable Text**
- **Selection Tree**
- **Text**
- **VolumeCostList**
- **VolumeSelectCombo**

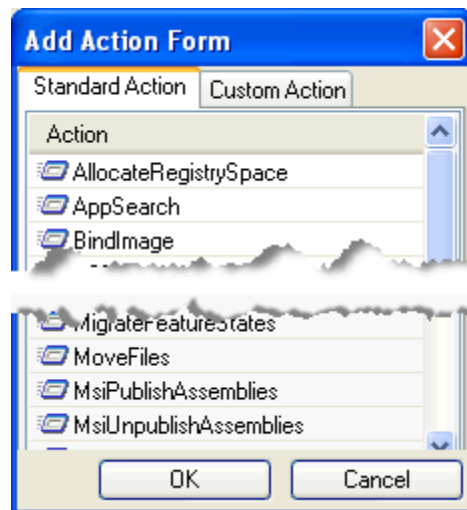
Event Sequencing



Event Sequencing is the definition of the order of execution for the standard actions that control the installation process and display the user interface dialog boxes.

IDS provides access to the event sequencing tables and allows events to be added, removed or the order of execution of actions to be changed. The following tables are available for modification:

- InstallUISequence
- InstallExecuteSequence
- AdvtUISequence
- AdvtExecuteSequence
- AdminUISequence
- AdminExecuteSequence

Within each table, click Add to add a new action to the table. An action is selected from a pre-defined list of Standard Actions and a user-defined list of Custom Actions. Custom Actions are defined in the Custom Actions pane.



Click Remove to delete an existing action from the table. Click  and  to reorder the actions in the table.

The execution of an action can be triggered to occur based on a condition. If the specified condition evaluates to True, the action or dialog is executed or displayed. The action or dialog is skipped if the expression evaluates to False. If there is no condition, the action or dialog is always executed or displayed.

Custom Actions

Custom Actions can be used to provide extra functionality to the features that Windows Installer provides. For example, custom actions can be used to call an executable, call a function in a DLL, run a VB script or set a property. Custom Actions are defined for use in the Event Sequencing tables.

Custom Action Name

Enter the name of the action. This name should not conflict with any pre-defined action name. If it does, the custom action will never execute.

Custom Type

Select the type of action from the drop list. Select from *VBScript stored in Custom Action table*, *VBScript stored in Binary Table*, *VBScript installed with Product*, *VBScript on Target machine*, *JScript stored in Custom Action table*, *JScript stored in Binary table*, *JScript installed with Product*, *JScript on Target machine*, *Dll stored in Binary table*, *DLL installed with Product*, *EXE stored in the Binary table*, *EXE installed with Product*, *EXE on target machine*, *Display Error Message*, *Set Property Value*, *Install MSI from Destination*, *Install MSI from Installation*, *Install MSI from Relative Path*.

Execution Options

Select from *Immediate Execution*, *Deferred Execution (User Context)*, *Rollback Execution (User Context)*, *Commit Execution (User Context)*, *Deferred Execution (System Context)*, *Rollback Execution (System Context)* and *Commit Execution (System Context)*.

Return Processing

Select from *Synchronous Execution*, *Synchronous*, *Ignore Exit Code*, *Asynchronous*, *Wait for Exit Code* and *Asynchronous Execution*.

Scheduling Options

Select from *Always Execute*, *Run First Time*, *Run Once Per Process*, *Run Only if UI Sequence is Run*.

Wrap an Executable

This process is used to wrap installations which cannot be repackaged. Installations should not be repackaged are if they are installing settings which differ on different machines, or if they are updating windows protected system files. Examples might include windows updates and security patches. Changes should not be captured but rather the original installation or patch should be wrapped in a Windows Installer MSI file for deployment.

Recycle Bin

Whenever an element of the project is deleted, it is removed from the project and placed into the recycle bin. All items in the recycle bin may be restored to their original place within the project by right-clicking on the recycle bin list.

Validation

VALIDATION

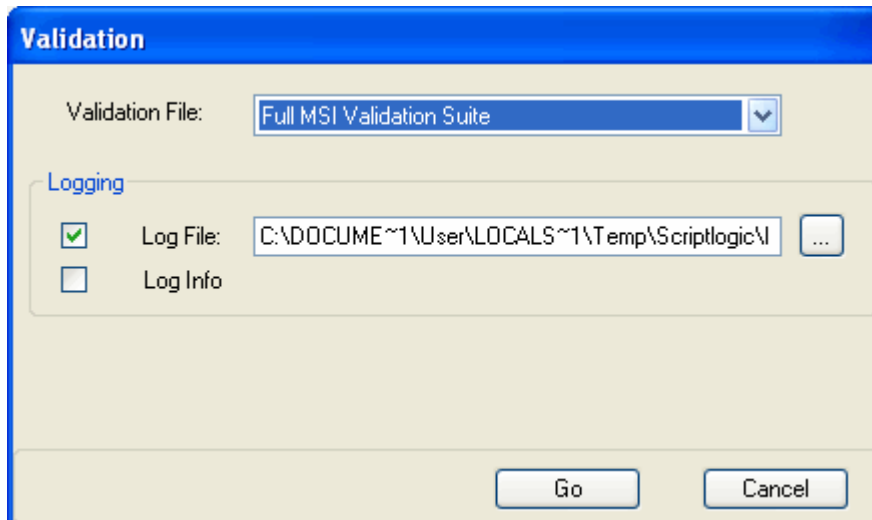
Validation validates the database that makes up the MSI. It checks everything from simple data checks like string length, data types and foreign keys to Windows Installer specific things like attributes of features.

MCE Validation

MCE validation is MSI Studio's faster implementation of Microsoft's ICE (Internal Consistency Evaluators) validation. MCE more rapidly determines errors with greater efficiency and links the errors directly to the MSI table entries for faster fixing of the problems.

ICE Validation

ICE Validation used to verify MSI packages. ICE is comprised of a set of rules that must be followed in order to have a viable package.



Validation File

Select the validation rules to use from the drop list. Select from *Full MSI Validation Suite, Windows 2000 Logo Program Suite, Windows XP Logo Program Suite, Merge Module Validation Suite.*

Logging

Log File

Select this box to create a log file. Specify the path and filename of the log file to hold the validation output.

Log Info

Select this box to turn on logging. The log file will contain all warnings, errors and sequencing information.

Tools

MSI STUDIO TOOLS

What is a Transform?

Windows Installer Transform file contains a list of modifications to be made to the MSI file during installation of the application. Transforms can be used to provide information to dialogs where user supplied information is required.

Transforms are typically used in two situations:

1. When a vendor-supplied MSI file is received and it needs to be customized. It is recommended that a vendor supplied MSI never be directly modified. Therefore, a transform file should be created to modify the package at run time.
2. When there is an MSI file which needs to be modified slightly for two different departments in an organization. Instead of creating two MSI files, create one package and two transforms which provide the modifications.

Create a New Transform

SELECT BASE MSI

Enter the path and filename of the base MSI file for which the transform file will be based upon. This is the MSI that is to be configured for a particular group of users, or a vendor MSI which is to be configured for your systems prior to deployment.

HOW TO CREATE TRANSFORM

There are three different ways which you can create your transform.

Create a transform directly based upon an MSI .

Select this option if changes need to be made to the MSI and know what these changes are. A typical scenario might be to add some registry keys which are required by your organization, or to set a property required by a particular department.

This option will open the base MSI and allow you to make changes to it. When you save the project those changes will be saved to your transform file. The base MSI will not change.

Create a response transform (Customize a Vendor MSI).

This option is used to capture the selections and configurations that are made during the UI sequence of an MSI.

For instance, if you have received an installation program from a vendor which is in the form of an MSI, select this option to create a transform. Changes that might be made during the installation UI sequence are to specify the directory to install the application to, what features to install and how (advertised, run from network, install locally) and other options such as not to run the application after installation. The configurations you make depends on what options are given during the UI phase of the base MSI.

Create a transform from the difference of 2 MSIs

This option is given for those repackagers who wish to create transforms from the differences in 2 MSIs. Often it is easier to maintain the base MSI and add changes to a copy of it. Before release or distribution you can create a transform from the 2 MSIs you have.


This option can also be used as a tool to find the differences between 2 MSIs.

Select the second MSI (the one containing the changes) in the entry. This can be done either by browsing or by entering the full path.

When the next button is selected a transform will be created and if successful will be opened along with the base MSI so as you can make further modification or check the changes. When finished save the transform.

SAVE AS

Enter the path and filename for the newly created transform file.

Click  to browse to the path where the file should be created.

GO

Click **GO** to create the transform.

Create a Response Transform

A response transform is used to capture the selections and configurations that are made during the UI sequence of an MSI.

For instance, if you have received an installation program from a vendor which is in the form of an MSI, select this option to create a transform. Changes that might be made during the installation UI sequence are to specify the directory to install the application to, what features to install and how (advertised, run from network, install locally) and other options such as not to run the application after installation. The configurations you make depends on what options are given during the UI phase of the base MSI.

Before you create a response transform:

- Make sure the application is not installed on the current machine.
- If the MSI with the transform will be deployed to multiple machines, make sure that the configuration of the machine you are working on is the same as those it will be deployed to.

Press the **GO** button when you are ready:

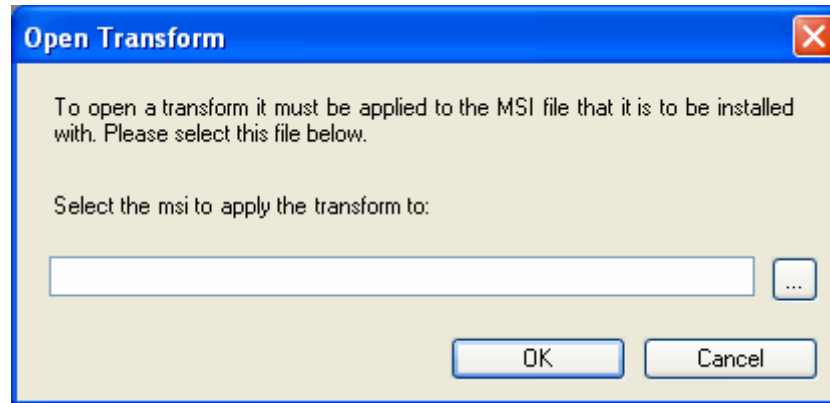
- The UI Sequence of the installation will now be stepped through – NOTE the application will not be installed.
- Configure the application as you want it deployed.


After the UI section of the installation has run the base MSI and the newly created transform will be opened. If you need to make any further changes then do them before saving the transform.

Editing an Existing MST

An existing MST file (transform) can be modified by selecting **Open** from the **File** menu. From the File Open dialog, browse to the MST file and select it. Click **Open**.

After it has been selected (and if the file has the extension .MST) the Open Transform dialog box will be displayed.



Select the MSI file to apply the transform to. Click  to browse to the MSI file. Click **OK** to confirm your selection.

The transform file is opened in Table View and is ready for possible modifications.

MSI STUDIO RUN WIZARD


The MSI Studio Run Wizard is useful to run MSI installations with certain command line parameters. It allows you to apply transforms to the MSI, run it silently, uninstall it, and create and open log files. It saves finding the correct command line syntax and running it from a command window.

The MSI Studio Run Wizard can be launched from the Tools menu or from Windows Explorer.

The screenshot shows the 'Desktop Authority MSI Studio Run Wizard' dialog box. It is divided into three main sections: 'WHAT', 'WHERE', and 'HOW'.
WHAT:
- 'Select the msi file to run.' with a text box and a browse button (...).
- 'Select the transforms to apply to it.' with a list box, a plus button (+), and a minus button (-).
WHERE:
- Radio buttons for 'Run the installation on the local machine' (selected) and 'Run the installation on a foreign machine'.
- 'Select the machine to run the msi on.' with a dropdown menu and a 'Test' button.
- 'Username:' and 'Password:' text boxes.
HOW:
- 'Select the command line options you wish to apply to your msi.'
- Checkboxes for 'Run the installation silently (/qn)', 'Uninstall the application (/x)', and 'Create a log file for the installation (/l)'.
- 'Enter the name of the log file to save to.' with a text box, a 'Create Temp File' button, and a browse button (...).
- Checkboxes for 'Open the log file after the installation has finished' and 'Command Line:' with a text box.
At the bottom are 'Help', 'GO', and 'Cancel' buttons.



WHAT

Select the MSI file to run

Enter the path and MSI file name you wish to run. Click  to browse to the MSI file. If the run the wizard is launched from the editor, and there is a project currently open, the name of the file will be placed in this entry.

Select the Transforms to apply to it

If one or more transform files are needed, enter the filename here.

Click  to add a file to the transform list. Click  to remove a transform file from the list. The transforms in this list will be applied to the MSI file before it is run.

WHERE

You can run the MSI on either the local machine or a foreign machine. When running the MSI file on a foreign machine it can only be run silently. However, this option is useful for testing a repackaged application. To run it on a foreign machine, enter the name of the machine. Enter the Username and Password in the appropriate fields and then click **Test** to make sure there is a viable connection to the machine.

HOW

Run the installation silently

Select this box to run the installation silently. The command line option `/qn` will be appended to the command line and the MSI will run with no user interface.

Uninstall the application

Select this box to uninstall the application. The command line option `/x` will be appended to the command line and the MSI will be uninstalled. This should only be used for applications that are already installed on the target machine.

Create a log file for the installation

A log file for the installation can be created by selecting this box. Once this is selected, you are then given the option of specifying the name of the log file and whether to open the log file after the installation has finished. The log file will be opened with notepad. To create a temporary file in the Windows temp directory, click the *Create Temp File* link.

Command line

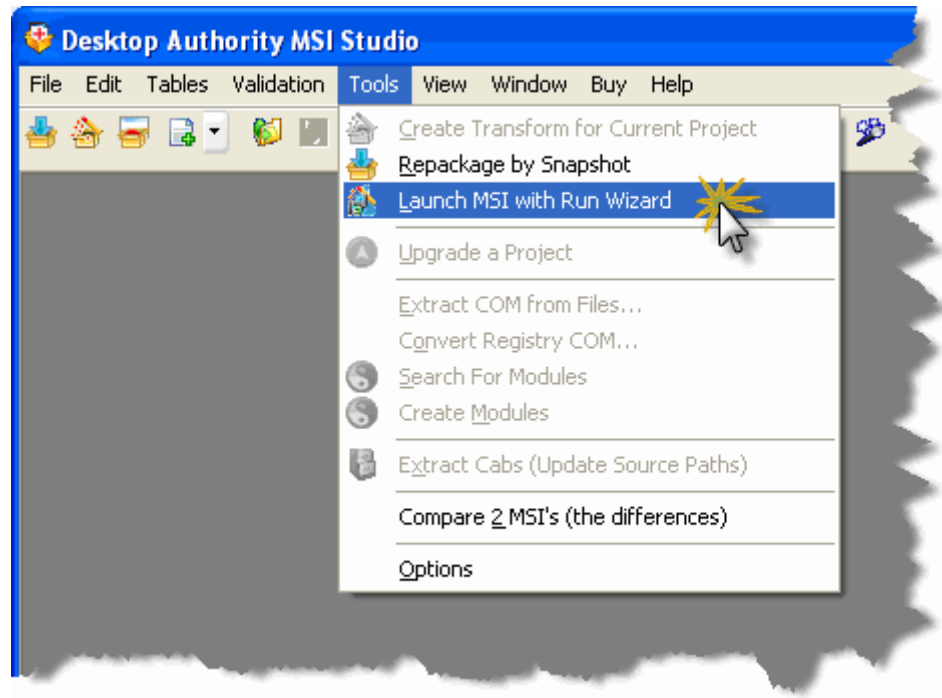
Enter any command line parameters to be used when running the installation.

GO

Click GO to start the Run Wizard.

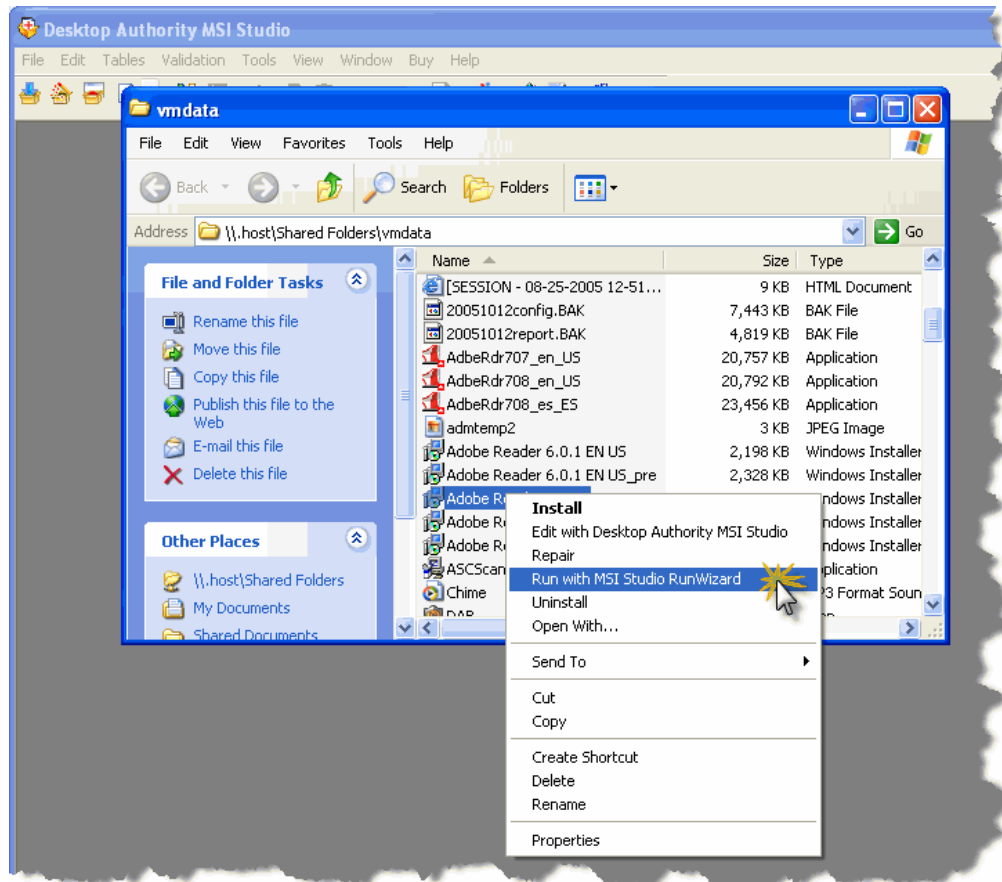
Launching the Run Wizard from within MSI Studio

To run the Run Wizard from within MSI Studio, click the  toolbar button or select *Launch MSI with Run Wizard* from the Tools Menu.



Launching the Run Wizard from Internet Explorer

Right click on the MSI file you wish to launch from within Internet Explorer and select **Run with Installer Design Studio Run Wizard**.



UPGRADE A PROJECT

If a project requires any changes an upgrade can be created for deployment to all machines that have a version of the application already installed. Upgrades can range from minor file changes or registry entry updates to a complete full blown application update that includes new features.

Upgrade

One off Upgrade

Upgrade the current project.

Type of Upgrade:

Small - (package code changed)

Minor - (package code and product version)

Major - (package code, product code and product version)

Current Version: New Version:

Automatic Upgrades

Automatically match the project version to the version of a file in the project.

File Whose Version to Read From:

...

Type of Upgrade: (every time the file version changes this type of upgrade will take place)

Small Minor Major

OK Cancel

One off Upgrade

Upgrade the current project

Select this box to upgrade the currently opened project. This may or may not affect the product version.

Type of Upgrade

Small

A small upgrade makes changes to application files that are very minor and do not require the product code to changes. This type of upgrade is commonly known as a Quick Fix Engineering (QFE) update. Typically a small upgrade will change one or two files or a registry key entry.

Minor


A minor upgrade makes changes to resource files. A minor upgrade can be used to add new features and/or components. A typical minor upgrade will include fixes that were previously released, but wrapped up into a patch. This type of upgrade is normally know as a service pack (SP) update. This upgrade will change the product version. Enter the new version in the provided entry box. The current version number is provided for reference.

Major

A major upgrade will provide a full update of a product. Typically this type of upgrade will remove the software from the machine and then install a new version. This upgrade will change the product code and version. Enter the new version in the provided entry box. The current version number is provided for reference.

Automatic Upgrades

Automatically match the project version to the version of a file in the project

Select this box to upgrade the project version based on a file included in the project. Click  to browse for and select the file in the project. Click OK to select the necessary file.

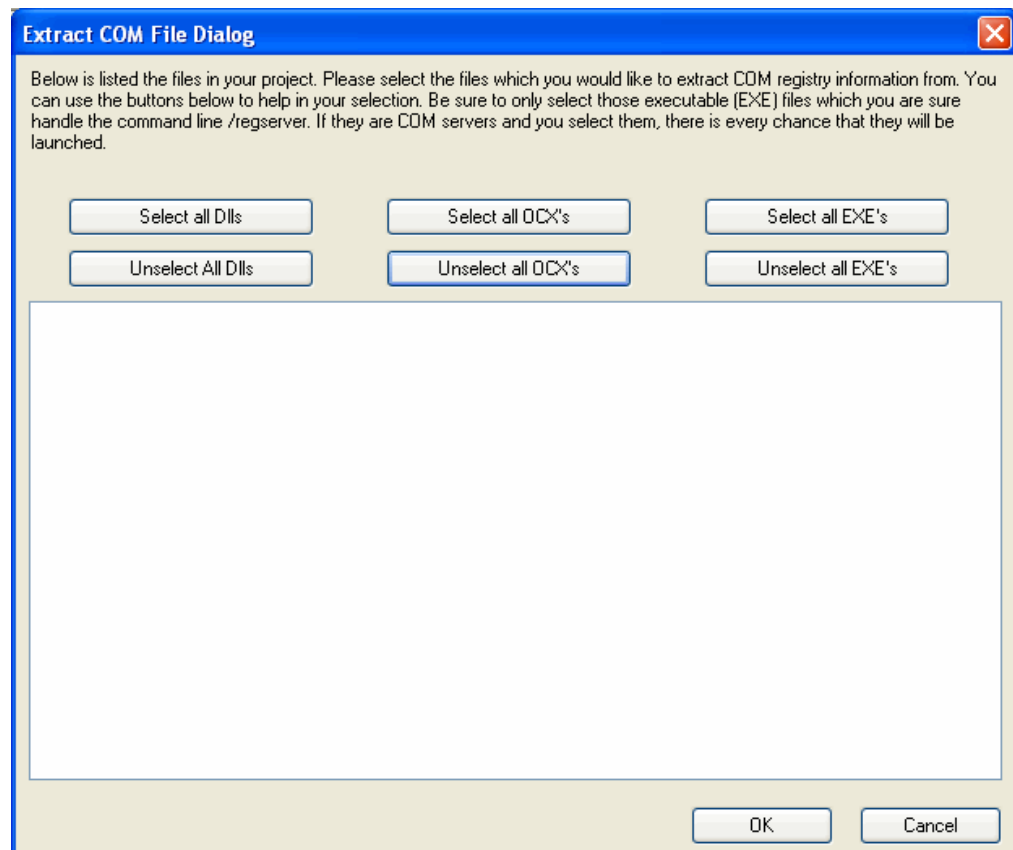
Type of Upgrade

Select either Small, Minor or Major as the type of upgrade to take place.

EXTRACT COM FILE FILES

All files in the project will be listed in this dialog. Select the files from which COM registry information should be extracted for. For each selected file, the COM registry information is extracted and placed into the registry table.

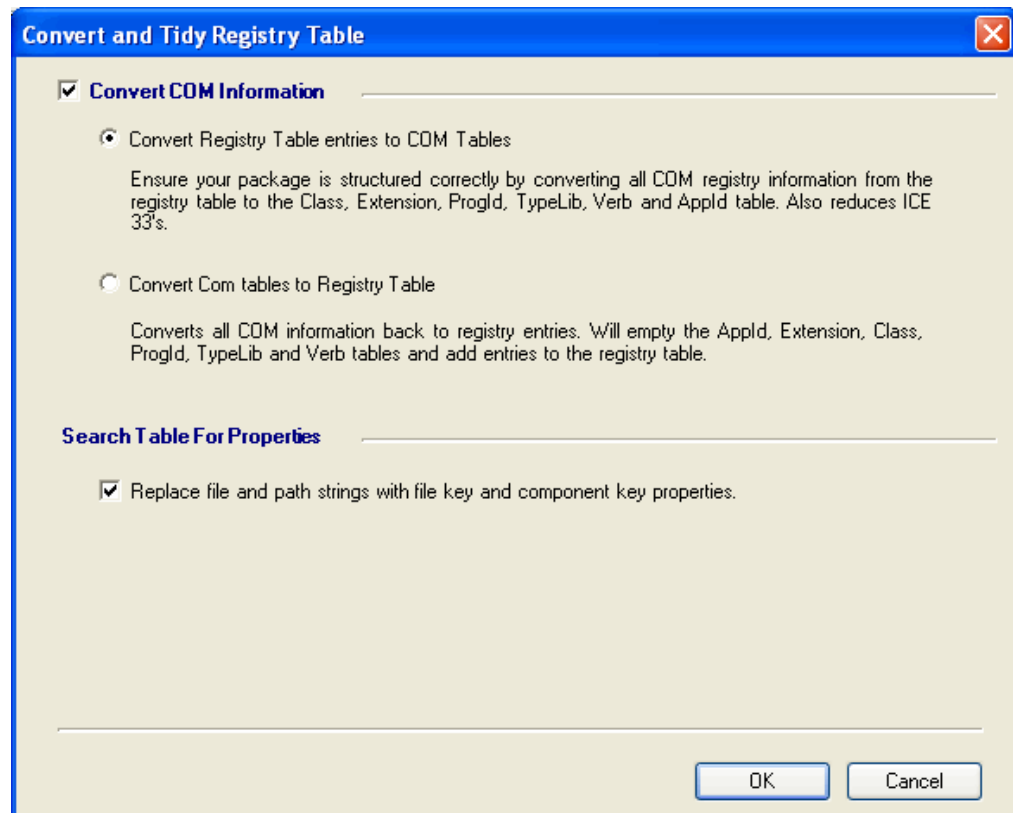
If the COM information is not extracted to the project, a custom action will need to be created for each COM object that calls regsvr32.exe to register the object.



Click a button on this dialog to select or unselect a specific group of file types. Click OK to start the COM extraction process.

CONVERT REGISTRY COM

COM registry information can be accessed in an MSI package in two different ways. They can be stored in the registry table or the



Convert COM Information

Convert Registry table entries to COM tables

Select this option to move all Registry table entries into the COM tables. This option is normally used if the package will support certain advertising features.

Convert COM tables to Registry table

Select this option to move the COM table information to the Registry tables.

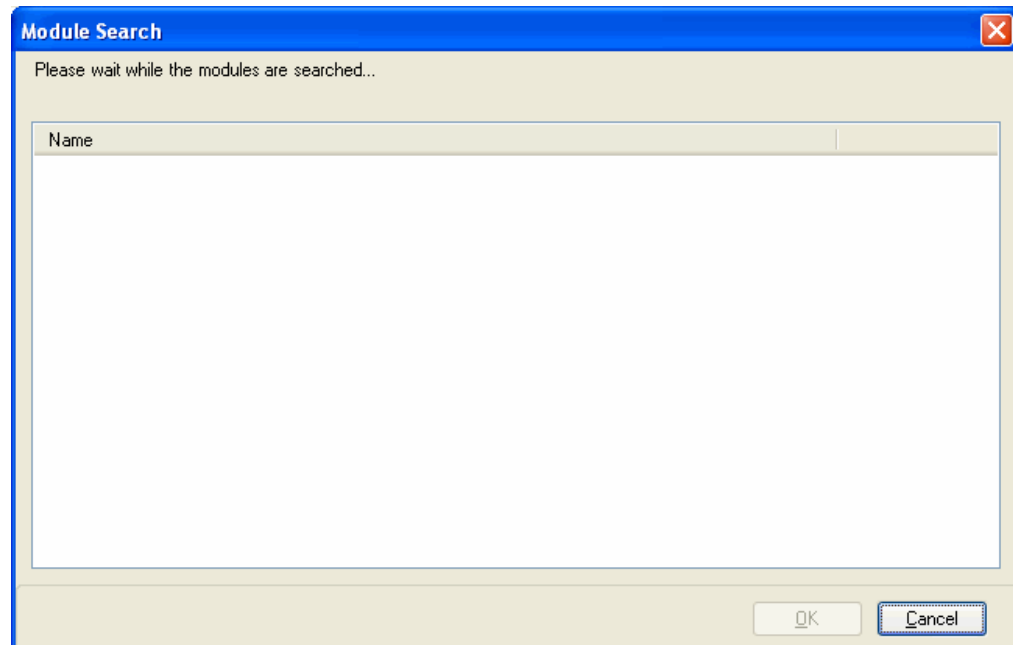
Search Table for Properties

Replace file and path strings with file key and component key properties

When reading COM tables or Registry table information, this option will replace the file and path entries with key and component key properties.

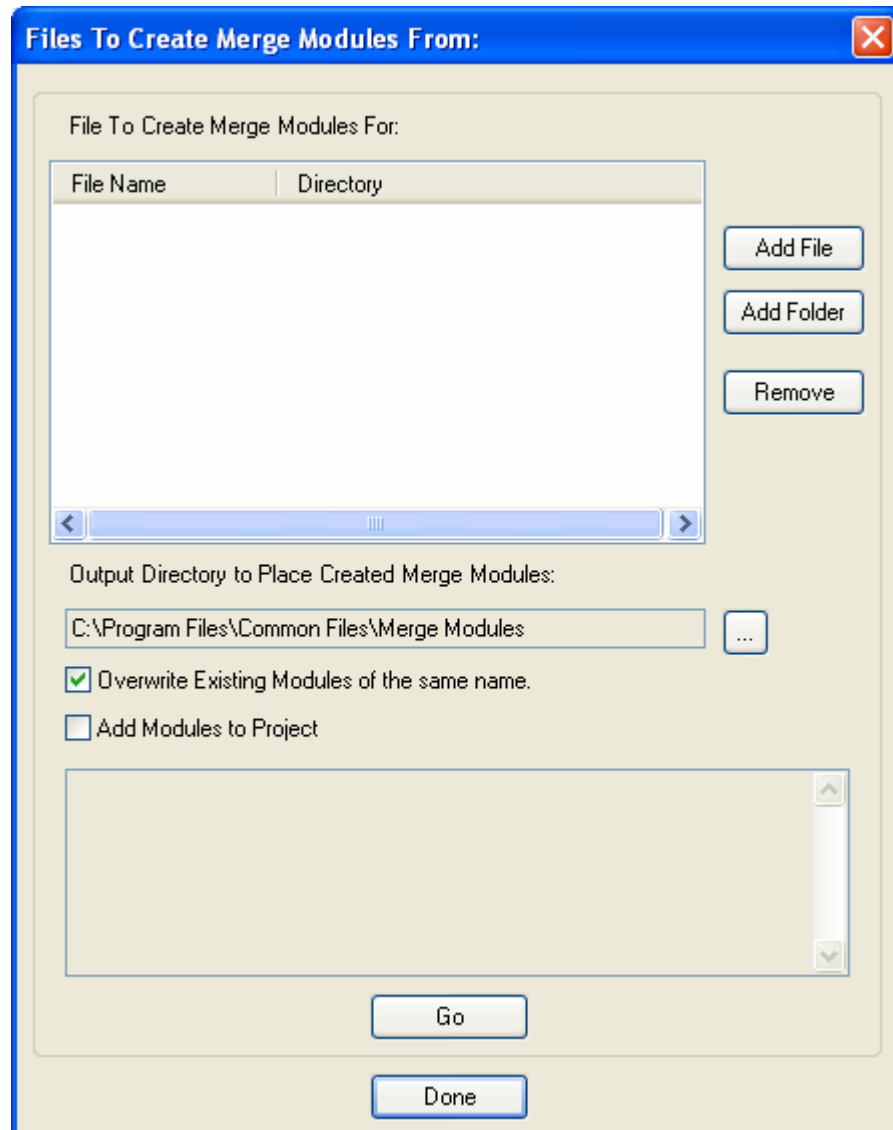
SEARCH FOR MODULES

This tool will search all of the files in the project and try to match them against the merge modules found in the defined search directories. When the search is complete, a dialog will prompt for the possible replacement of these files with the merge files. Search paths for merge modules are created by selecting the Tools > Options menu. Select the Merge Modules tab. All found merge modules will be listed.



CREATE MERGE MODULES


This tool is used to create Merge Modules. Merge modules are files that contain shared code, files, resources, registry entries and setup logic combined together in a single file. This file can then merged into the MSI project.



Files to Create Merge Modules for

This list contains all files that will be contained into the resulting merge module. Click **Add file** to select a file from within your current project to add to the merge module. Click **Add folder** to select a folder from within your current project to add to the merge module. Click **Remove** to remove an entry from the list.

Output Directory to place created Merge Modules

Click  to browse to and select an output folder for the new merge module.

Overwrite Existing modules of the same name

Select this box to overwrite an existing merge module in the output folder if one exists.

Add Modules to project

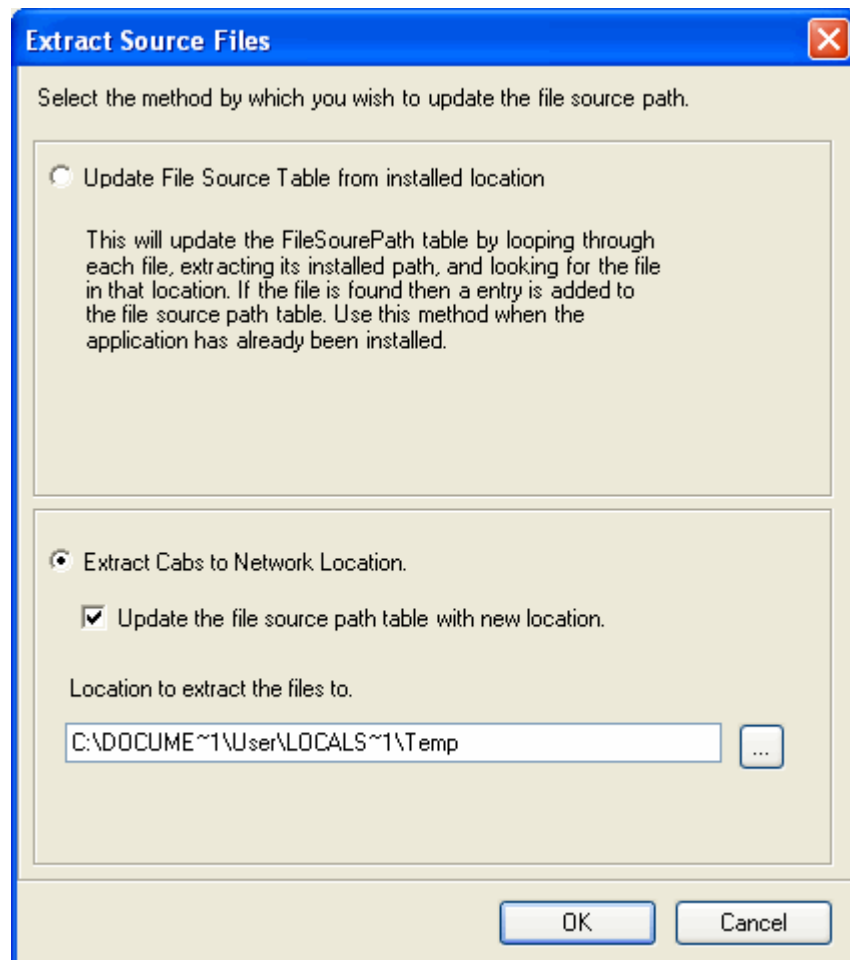
Select this box to add the newly created merge module to the currently opened project.

Go

Click Go to start the creation of the merge module.

EXTRACT CABs

Use this tool to extract all the files from the cab files inside (or outside) the MSI file and place them in a sub directory. The path to these files are then added to the File Source Table. This table is used to store the source directory of files in the project. The source directory is used when the project is built. When the project is built after this process, it will use these extracted files.



Update File Source Table from installed location

Once the project is installed, select this option to update the File Source Table. Each file in the project will be verified in its location and the File Source Table will be updated with the path.

Extract Cabs to Network Location

Select this option to extract the cab files from the current MSI file.

Update the file source path table with new location

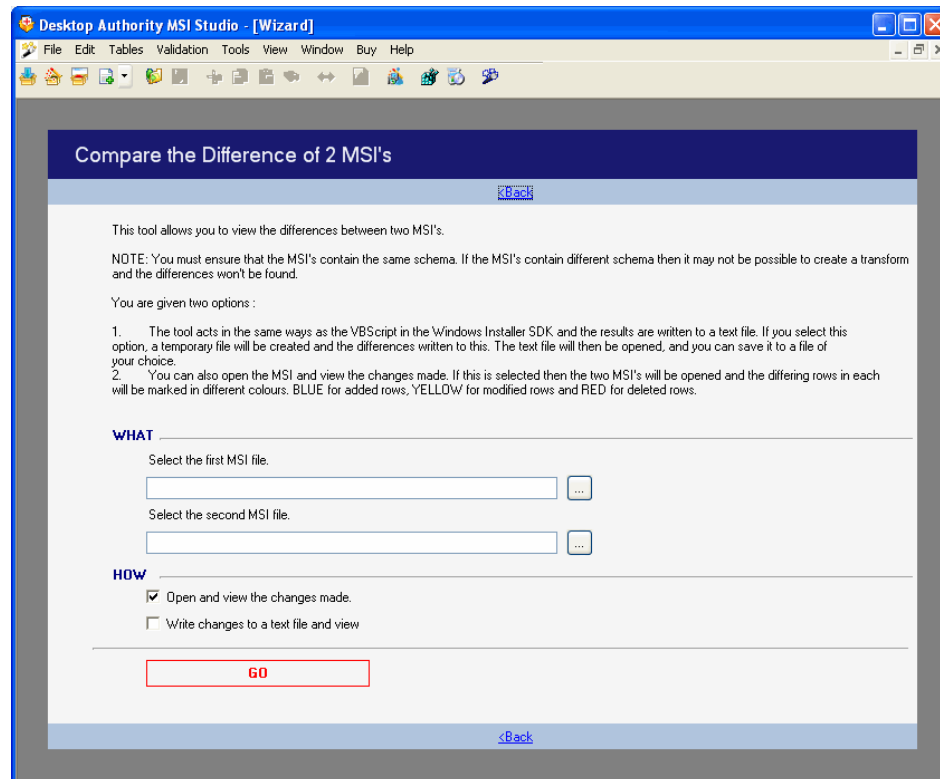
Select this box to update the File Source Table with the location of the files once they are extracted.

Location to extract the file to


Specify the path where the cab file will be extracted to.

VIEW THE DIFFERENCE BETWEEN 2 MSIS

MSI Studio contains a tool to compare two MSI files. The process analyzes the tables that make up the MSI files and displays the results in a table view with color coded rows or in a text file.



What

Enter the path and filename of each file to be compared. Click  to browse to the MSI file.

How

Open and view the changes made

Select this box to view the comparison differences in a color coded table view.

Write changes to a text file and view

Select this box to create a text file containing the results of the file comparison. After the comparison is complete, this text file will be opened for viewing.

Results

*Directory_	*Component_
INSTALLDIR	Registry_eBook
INSTALLDIR	Registry_ENU_Product
INSTALLDIR	Registry_ENU_User
INSTALLDIR	Registry_Reader_Updates
INSTALLDIR	Registry_Optimization
INSTALLDIR	Registry_ProgID_AcroExch.Document_NotInsertable
INSTALLDIR	Registry_DisableBrowserCheck_Acro6
INSTALLDIR	EULA_Accept_Registry_R
INSTALLDIR	Registry_ProgID_AcroExch.Document_Shell
INSTALLDIR	Registry_URL_Handler
INSTALLDIR	Registry_Main_User
INSTALLDIR	Comp_Reader
INSTALLDIR	Registry_Installer_ReaderBig
BROWSER	Registry_Netscape_User
CACHE_SETUP_FILES	Reader_Bin_AcroRd32.exe
RDR_BIG	Help_Full_ENU
RDR_BIG_ENU	Help_Full_ENU
RDR_BIG_ENU_705	Help_Full_ENU
RDR_BIG_ENU_707	Help_Full_ENU
INSTALLDIR	Registry_Vista_ElevationPolicy
RDR_BIG_ENU_708	Help_Full_ENU

Differences in Table View

Results Legend

Yellow - Modified Row
 Red - Deleted Row
 Light Blue - Added Row

```

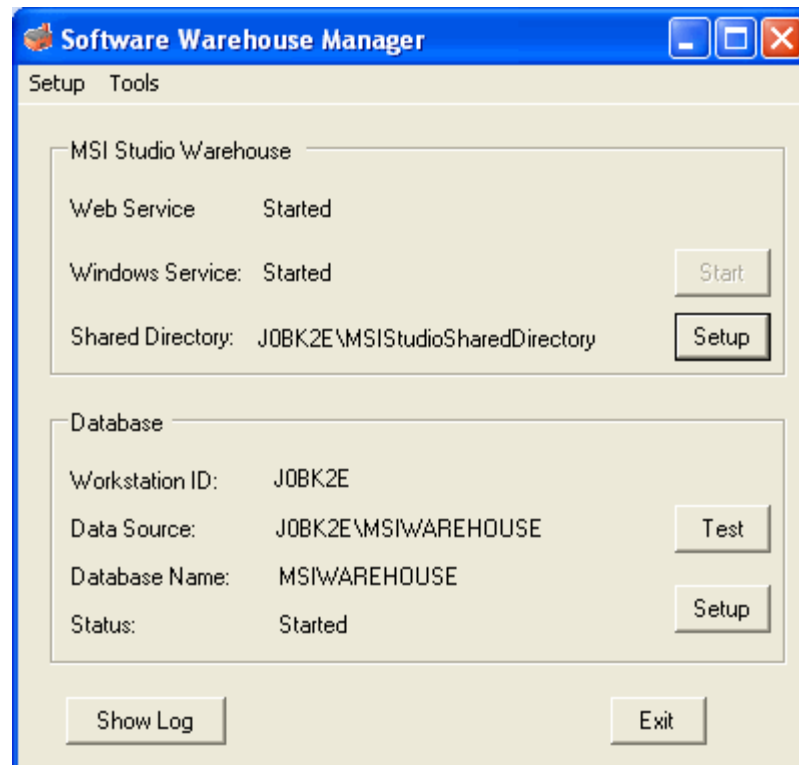
AppSearch INSERT [PREV_READER_PATH_NO_FILENAME, _PreviousReaderPathNoFilena
AppSearch INSERT [RDR_708_PATCHED_FOUND, _Rdr_708_Updates_Key]
Property Value [ERROR_PATCHED_FOUND_ABORT]{}->{This version of Adobe Rea
Property Value [BACKUP_PREFERENCES]{}->{YES}
Property Value [BackupPreferencesKey]{}->{Software\Adobe\Backup\Preferen
Property Value [PRESERVE_INSTALLDIR]{}->{YES}
Property Value [ERROR_PROCESS_LOCKED_ABORT]{}->{A process is running tha
Property Value [ProductCode]{AC76BA86-7AD7-1033-7B44-A70700000002}->{{
Property Value [ProductName]{Adobe Reader 7.0.7}->{Adobe Reader 7.0.8}
Property Value [ProductVersion]{7.0.7}->{7.0.8}
Property Value [BrandName]{Adobe Reader 7.0.7}->{Adobe Reader 7.0.8}
Property Value [OpenWith]{Open with Adobe Reader 7.0.7}->{Open with Adob
Property INSERT [ERROR_PATCHED_FOUND_ABORT]{open with Adobe Reader 7.0.7}
Property INSERT [BACKUP_PREFERENCES]{open with Adobe Reader 7.0.7}->{Open
Property INSERT [BackupPreferencesKey]{open with Adobe Reader 7.0.7}->{Op
Property INSERT [PRESERVE_INSTALLDIR]{open with Adobe Reader 7.0.7}->{ope
Property INSERT [ERROR_PROCESS_LOCKED_ABORT]{open with Adobe Reader 7.0.7
Binary Data [SetAllUsers.d11]{}->{Binary.SetAllUsers.d11}
Binary Data [AdobeIsf]{}->{Binary.AdobeIsf}
Binary Data [ISSELFREG.DLL]{}->{Binary.ISSELFREG.DLL}
Binary INSERT [SetAllUsers.d11]{}->{Binary.ISSELFREG.DLL}
Binary INSERT [AdobeIsf]{}->{Binary.ISSELFREG.DLL}
Binary INSERT [ISSELFREG.DLL]{}->{Binary.ISSELFREG.DLL}
File Sequence [FL_at171_d11_4]->{x86.3643236F_FC70_11D3_A536_0090278A1
File Sequence [AcroIEHelper.d11]{263}->{262}
File Sequence [AcroPDF.d11]{264}->{263}
File Sequence [pdfshe11.d11]{242}->{241}
File Sequence [inppdf32.d11]{262}->{261}
File Sequence [FL_msvecr71_d11_3]->{x86.3643236F_FC70_11D3_A536_0090278
File Sequence [Adobelinguistic.d11]{286}->{285}
File Sequence [icudt261.dat]{283}->{282}
File Sequence [symbol.txt2]{279}->{278}
File Sequence [zdlingbat.txt]{278}->{277}
File Sequence [CORPCHAR.TXT2]{282}->{281}
File Sequence [ROMAN.TXT1]{280}->{279}
File Sequence [SYMBOL.TXT]{277}->{276}
File Sequence [CP1252.TXT1]{281}->{280}
    
```

Differences in Text File Export

SOFTWARE WAREHOUSE MANAGER

***The Software Warehouse is only available in the Professional Version of MSI Studio.**

The Software Warehouse Manager is a tool that is used to show the status of the various MSI Studio components. This includes the Web Service, Windows Service, Database and shared directory.



Setup

Shared Directory

MSI Studio requires a shared directory that can be used as a shared access point. Specify the share and shared name if it needs to be created. The specified share must have READ+Write permissions assigned to it.

Database

The Software Warehouse requires an instance of Microsoft SQL Server or a Microsoft SQL Server Desktop Engine (MSDE) installation. You may either select an existing SQL Server instance or the Warehouse Manager can install a new instance of MSDE.

Refresh

Select **Refresh** to refresh the Software Warehouse Manager dialog.

Exit

Select **Exit** to close the Software Warehouse Manager.

Tools**Options**

Select **Recalculate all conflicts each time a project is loaded** to find conflicts each time a project is loaded. Selecting this option will slow down the time it takes to load a project. If this option is not selected, conflicts can be detected by manually selecting this option in the Software Warehouse.

MSI Studio Warehouse

This section in the Software Warehouse Manager shows the status of the Web and Windows services.

Start

Click Start to manually start the Windows service.

Shared Directory Setup

MSI Studio requires a shared directory that can be used as a shared access point. Specify the share and shared name if it needs to be created. The specified share must have READ+Write permissions assigned to it.

Database

The Software Warehouse requires an instance of Microsoft SQL Server or a Microsoft SQL Server Desktop Engine (MSDE) installation. You may either select an existing SQL Server instance or the Warehouse Manager can install a new instance of MSDE.

Test

Click **Test** to test the connection to the specified database.

Setup

Click **Setup** to configure the MSDE or SQL Server database.

Show Log

Click **Show Log** to open the MSI Studio folder containing troubleshooting log files.

Exit


Click **Exit** to close the Software Warehouse Manager.

OPTIONS

Template Options

A template is a base MSI file that contains standard entries such as dialogs, standard properties and error messages. Templates are then used as the starting point when creating a new MSI. New entries are added to the template to make the finished MSI.

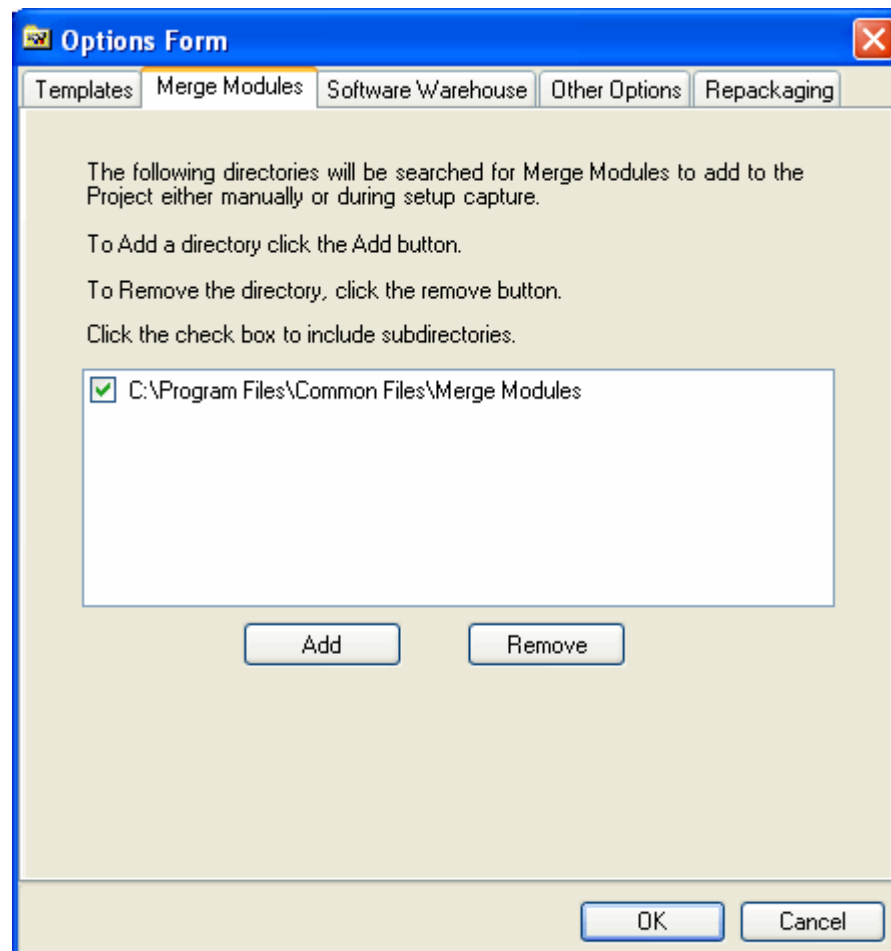
MSI Studio comes with two basic templates -- Repackage Template and Windows Template. If neither template suits your needs, they may be customized or a new template created.

The Template Options dialog (Tools > Options > Templates menu) configures where template files are stored. The templates folder, by default, is created on the local machine in the \Documents and Settings\User\Application Data\Scriptlogic\IDS\templates\ folder. To change this default, enter the new template folder or click  to browse to it.

Merge Modules

Merge Modules are files that contain code that delivers Windows Installer components. Shared code, resources, registry entries and setup logic to applications are contained within a merge module. When a merge module is used in an MSI, all of its resources are incorporated into the MSI file, leaving the merge module no longer being needed.

The Merge Modules options provide a dialog in which paths are specified. These paths will be searched for Merge Modules when the Merge Modules panel is selected in the IQ Views of an open project.



***The Software warehouse is only available in the Professional Version of MSI Studio**

Click **Add** to insert a folder into the search list.

Click **Remove** to remove a folder from the search list.

Select the box to the left of the path to include all subdirectories of the folder in the Merge Modules search.

Software Warehouse

***The Software Warehouse is only available in the Professional Version of MSI Studio.**

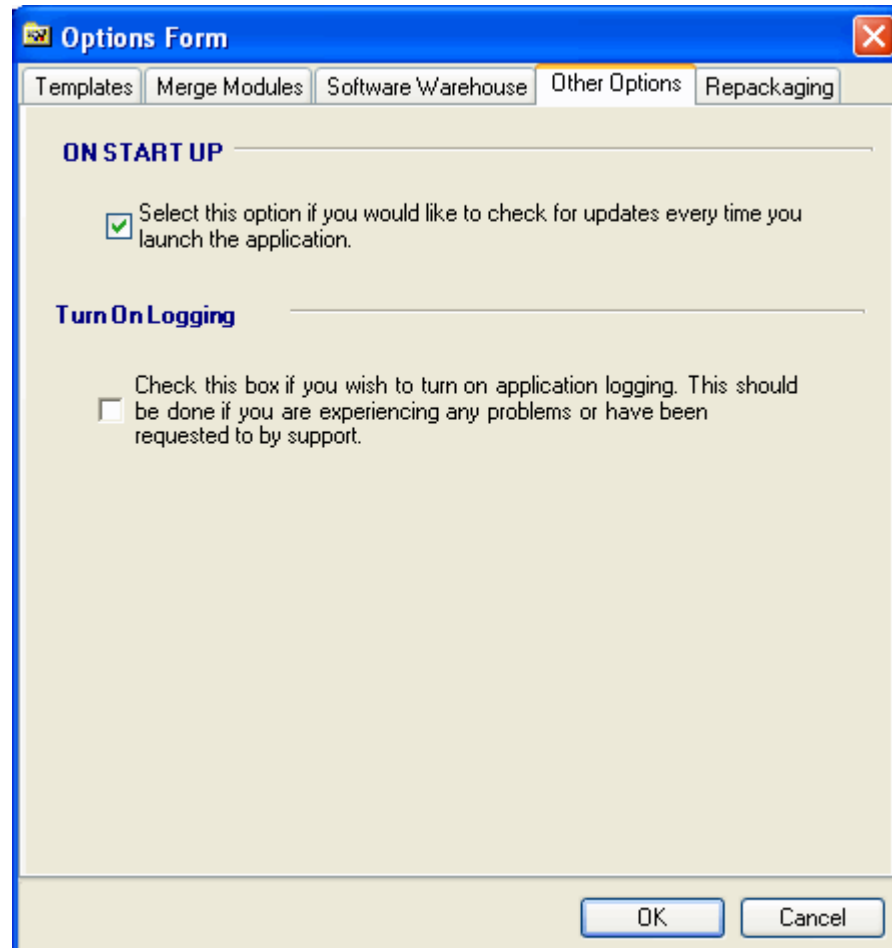
The Software Warehouse uses an SQL database to query MSI projects for Conflict Detection. The database can be either MSDE or SQL Server and is configured in the Software Warehouse Manager.

The Software Warehouse options tab declares the server that the Software Warehouse database exists on. Upon installation and configuration of the database, this option will be entered. If the database location is manually changed, be sure to change the server location here.

The screenshot shows a dialog box titled "Options Form" with a close button in the top right corner. It features five tabs: "Templates", "Merge Modules", "Software Warehouse", "Other Options", and "Repackaging". The "Software Warehouse" tab is selected. The main area contains the following text: "Enter the name of the server that the webservice has been installed on and is running. If it is on the local machine then just enter 'localhost'". Below this text is a text input field containing the value "localhost". At the bottom right of the dialog are two buttons: "OK" and "Cancel".

Other Options

Select the Other Options panel to set various miscellaneous options.



***The Software warehouse is only available in the Professional Version of MSI Studio**

On Start Up

Select this box to automatically check for program updates every time MSI Studio is launched.

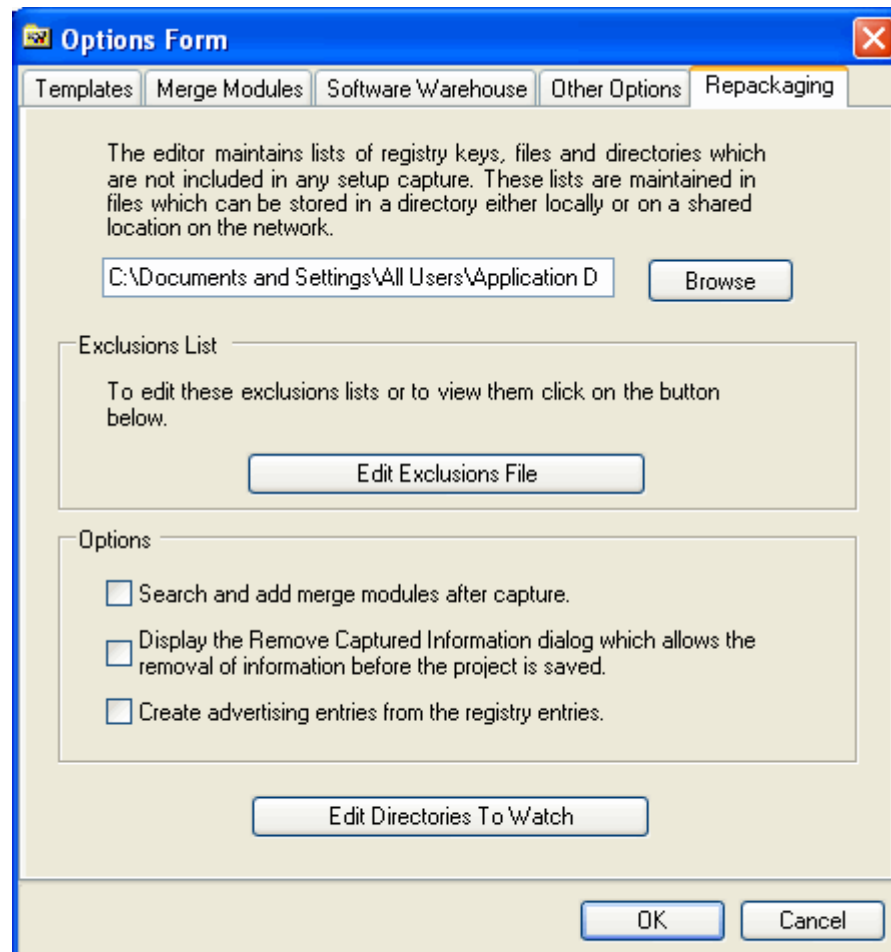
To manually check for MSI Studio updates select the Check for Updates.. item from the Help menu.

Turn On Logging

Logging is turned off by default when first installed. It is used as a troubleshooting tool. Debugging messages are logged to these files.

In the Software Warehouse, windows service and manager logging is turned on by default.

Repackaging



***The Software warehouse is only available in the Professional Version of MSI Studio**

Exclusion File Location

Enter the path to a folder where exclusion files are stored. Click **Browse** to locate the folder.

Exclusions List

Click **Edit Exclusions File** to define Files, Registry Keys and/or Directories to be excluded from the setup capture.

Options

Search and add merge modules after capture

Select this box to search for and add merge modules to the project after the capture is complete.

Display the Remove Captured Information dialog which allows the removal of the information before the project is saved

Select this box to display removed information before the project is saved.

Create advertising entries from the registry entries

Select this box to create advertising entries in the project from registry entries specified.

Edit Directories to Watch

Add or delete directories and drive that will be searched for changes. All subdirectories will be automatically included.

Build Options

Build Type

Files can be compressed into cab files to make a smaller footprint. These cab files can be placed inside the MSI, external to the MSI or not compressed into a cab file but placed into the MSI file. Select the appropriate option from the drop list.

Cab Size

If the external cab file build type is selected, the cab file can be made to fit on different types of media. Select the type of media that will be used for the cab files. Select from No Limit (size does not matter), 1.44 Floppy Disk or CDROM.

Bootstrapper

Create a setup.exe to call the MSI

Select this box to create a setup.exe file to run the MSI. The setup.exe file will exist separately from the MSI.

Create a self-extracting MSI with all the files inside

Select this box to create a self-installing MSI package which contains all of the information necessary for the installation of the package.

Include the Windows Installer Runtime

Select this box to include the necessary files to install Windows Installer on the target computer if it does not already exist.

Digital Signature Option

Windows Installer uses digital signatures to detect corrupted files.

Select **Create a digital signature** on save to create a signature file when a project is saved. When creating the digital signature specify a Name, Credential File and Private Key File.

Software Warehouse

OVERVIEW

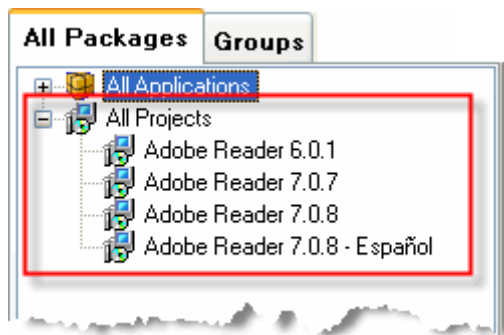
***The Software Warehouse is only available in the Professional Version of MSI Studio.**

The Software Warehouse is a repository that holds the details of imported Windows Installer applications. The Software Warehouse provides Conflict Detection and Resolution by storing all details of specified applications in the repository. Applications in the Warehouse can also be run through a Pre-deployment testing process.

The Software Warehouse categorizes Installer packages as Projects, Applications and Groups.

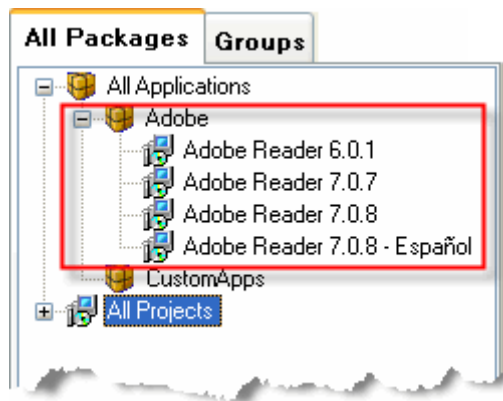
What is a Project?

A **Project** in the Software Warehouse is a complete MSI Package. This includes the MSI file as well as any associated transform (MST). Projects are imported into the Warehouse repository by selecting *Import Project (.msi file)* from the Warehouse menu. When the project is imported, the project name is extracted from the MSI file. Conflict Detection can be run on a specific package against a selected group or all packages.



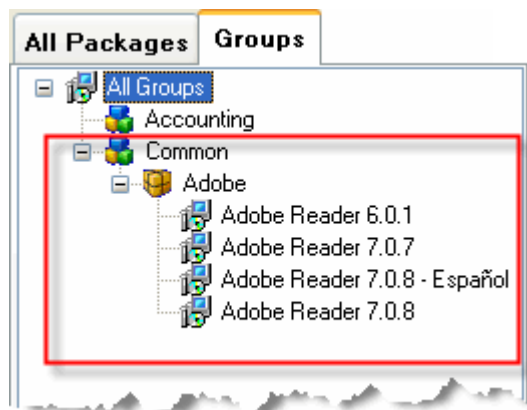
What is an Application?

An **Application** is a container that holds one or more Windows Installer packages in the Warehouse. These applications are related by program type but differ in version. For example, an Application called *Adobe* may contain several projects for different versions of Adobe. When a project is imported into the Warehouse, a new Application name can be specified or an existing one can be chosen from the Application drop list.



What is a Group?

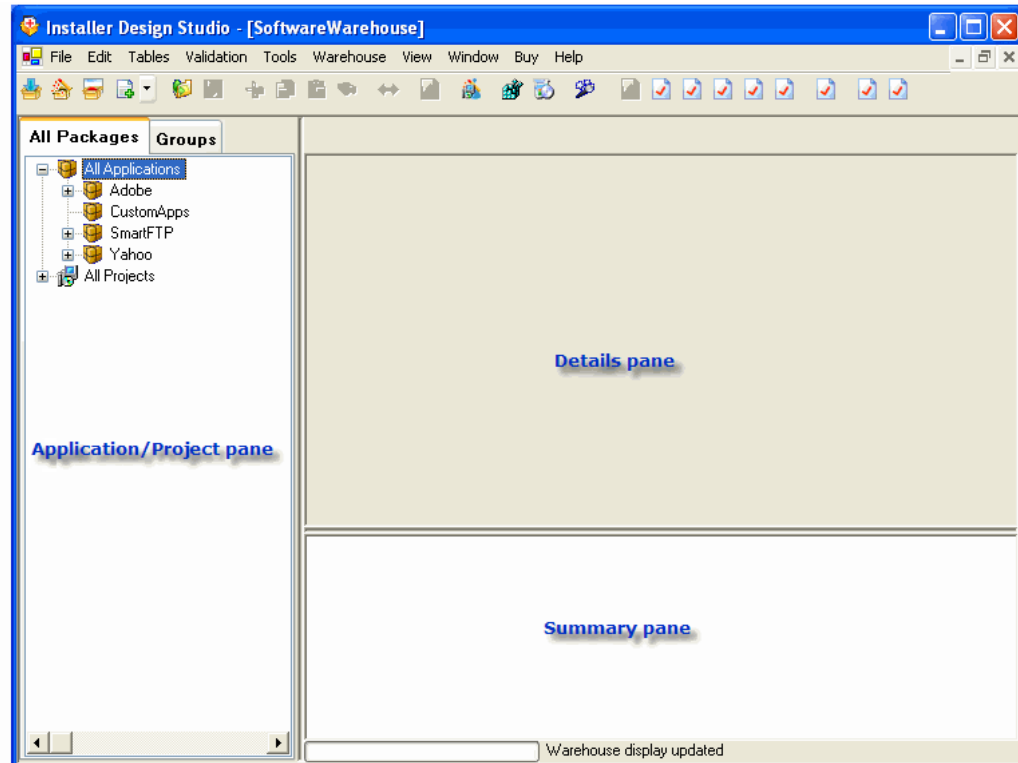
A **Group** is a container that holds related application. These applications are related in the way they will be installed. For example the Accounting Group may hold applications that are common to the Accounting department or the Common group can hold applications which are commonly distributed to everyone. Conflict Detection can be run across all packages within a selected group.



Using the Software Warehouse

*The Software Warehouse is only available in the Professional Version of MSI Studio.

The Software Warehouse dialog consists of three panes. They are the Application/Project pane, Details pane and the Summary pane.



The **Application/Project** pane consists of two tabs. These tabs group imported packages into visibly distinct and sections, Applications, Projects and Groups. Packages can be assigned to a specific Application when it is imported into the Software Warehouse. Packages can be moved into appropriate Applications or Groups at any time.

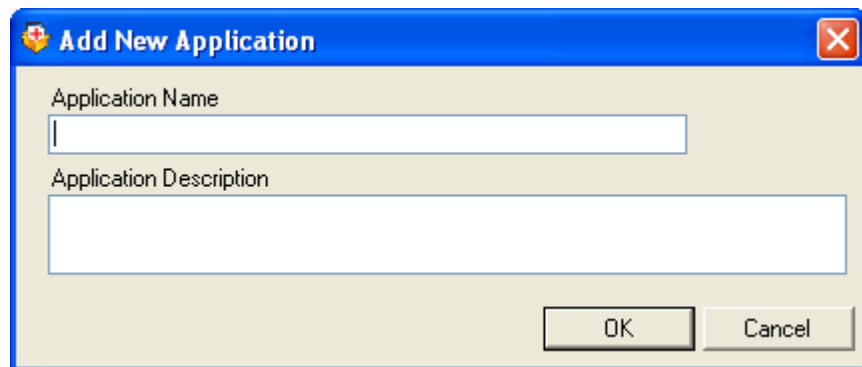
The **Details** pane displays the details of the selected Application, Group or Project. Click Update Details to save any changes made on the selected details dialog.

The **Summary** pane displays the status of processes as they run.

Managing Applications

Add New Application

To add a new application to the Application/Project pane, right-click on the All Applications or one specific application in the tree. Select *Add New Application* from the popup menu. This menu selection can also be accessed from the Warehouse menu.



Application Name

Enter the desired application name.

Application Description

Enter a description of the application.

Delete Application

To remove an existing application from the Application/Project pane, right-click on the specific application in the tree. Select *Delete Application* from the popup menu. This menu selection can also be accessed from the Warehouse menu.

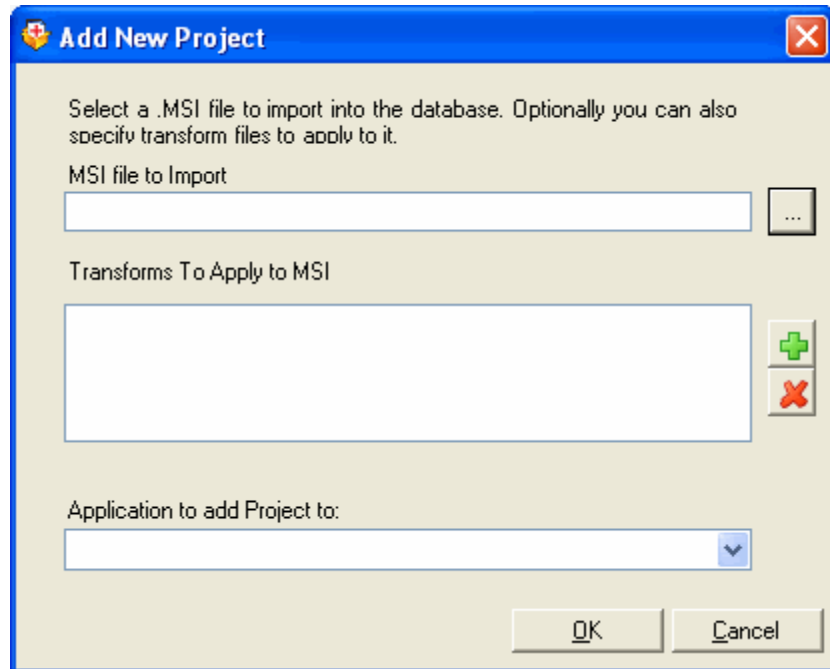
Add Existing Project to Application

To add one or more existing projects to an Application/Project, right-click on the specific application and select *Add Existing Project to Application* from the popup menu. Highlight one or more projects in the list and click OK. This can also be accessed from the Warehouse menu.


Managing Projects

Import project (.msi file)



To add a new project to the Application/Project pane, right-click on the All Projects or one specific project in the tree. Select *Add New Project* from the popup menu. This menu selection can also be accessed from the Warehouse menu.



MSI file to import

Enter the path and name of the project (MSI file) to import into the warehouse. Click  to browse to the MSI file.

Transforms to apply to MSI

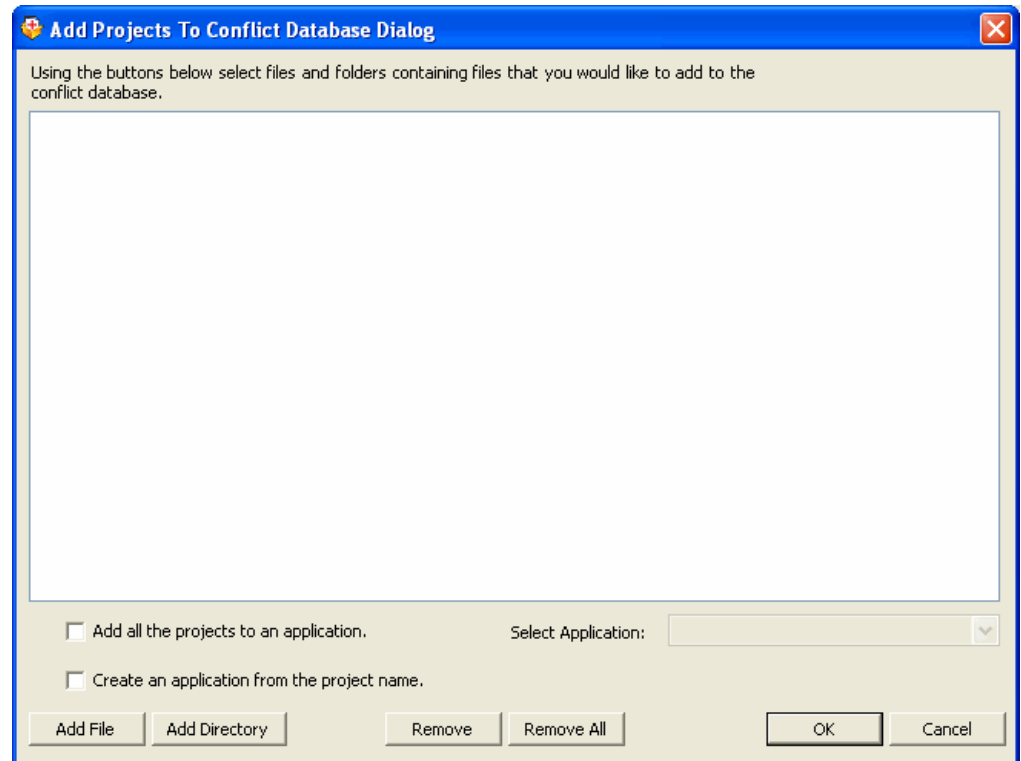
Specify one or more transform files (MST) that will apply to the project. Click  to browse to and select a transform file. Click  to remove a transform file from the list.

Application to add project to

Type a new application name into the entry or select an existing application name from the drop list.

Import multiple projects (.msi files)

Multiple Projects can be batch imported into the Software Warehouse.



Add File

Click **Add File** to browse to and select a project to add to the Software Warehouse.

Add Directory

Click **Add Directory** to browse to and select a folder containing one or more projects to add to the Software Warehouse.

Remove

To remove a project from the import list, highlight the project and click **Remove**.

Remove All

Click **Remove All** to empty the import list of all projects.

Add all projects to an application

Select this box to automatically add all projects specified in the import list to an application. Once this option is selected, select the application *Select Application* drop list.

Create an application from the project name

Select this box to create an application for each project that is imported. The application name is based on the project name.

Delete project

To remove an existing project from the Software Warehouse, right-click on the specific application in the project tree or within an application. Select *Delete Project* from the popup menu. This menu selection can also be accessed from the Warehouse menu.

Add Existing Project to Application

To add one or more existing projects to an Application, right-click on the specific application and select *Add Existing Project to Application* from the popup menu. Highlight one or more projects in the list and click OK. This can also be accessed from the Warehouse menu.

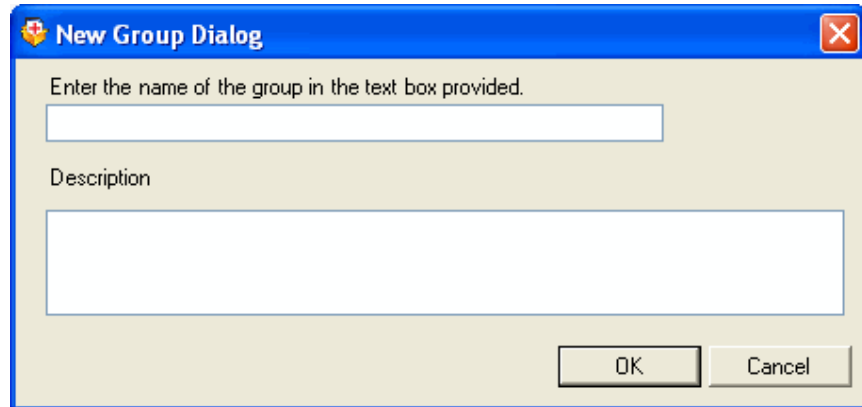
Remove Project from Application

To remove an existing project from an application, right-click on the specific project within an application in the project tree. Select *Remove Project from Application* from the popup menu. This menu selection can also be accessed from the Warehouse menu. The project will be removed from the application but will remain in the Software Warehouse database.

Managing Groups

Add New Group

To add a new group to the Group pane, right-click anywhere in the Group tab. Select *Add New Group* from the popup menu. This menu selection can also be accessed from the Warehouse menu.



Enter the name of the group

Enter the desired group name.

Description

Enter a description of the group.

Delete Group

To remove an existing group from the Group tab, right-click on the specific group in the tree. Select *Delete Group* from the popup menu. This menu selection can also be accessed from the Warehouse menu.

Remove Child Group

To remove an existing child group from the Group tab, right-click on the specific group in the tree. Select *Delete Group* from the popup menu. This menu selection can also be accessed from the Warehouse menu.

Add Application to Group

To add an application to a group, right-click on the group in the Group tab. Select *Add Application to Group* from the popup menu. This menu selection can also be accessed from the Warehouse menu. Select the application from the popup dialog and click OK.

Remove Application from Group

To remove an application from a group, right-click on the application in the Group tab. Select *Remove Application from Group* from the popup menu. This menu selection can also be accessed from the Warehouse menu.

Create Zero-Touch Testing MSI

Zero-Touch Testing is the act of creating a test package based on the project settings, where the package, when tested, does not actually install anything to the computer. Zero-Touch Testing is used to reduce failed deployments, by providing administrators and packagers with the ability to test a package before it is generally released. Often deployments fail because machines do not have enough free space, do not meet the system requirements of the package or have improper permissions or privileges for installing the application. Zero-Touch Testing creates an empty test package, based on the original package. The test package runs on destination machines, testing all necessary conditions, such as, disk size requirements, file and registry permissions requirements, launch conditions, and administrative privilege requirements. The results of the test deployment will be reported back to the warehouse.

Administrators and packagers can use the results of the test installation to find out where and why a package is likely to fail before it is actually deployed. This gives administrators time to correct the problems before the final package is deployed.

Recalculate ALL conflicts

Select this menu item to recalculate all conflicts for all applications in the Software Warehouse.

Refresh

To refresh the package tree, right-click anywhere on the tree. Select *Refresh* from the popup menu. This menu selection can also be accessed from the Warehouse menu.

Open this project

To open the highlighted project in the package tree, right-click on the project and select *Open this project* from the popup menu. This menu selection can also be accessed from the Warehouse menu.

Advanced

Right-click on the Package/Group tree and select *Advanced* options. This menu selection can also be accessed from the Warehouse menu.

Update

Update (if newer date)

Select this menu option to update the currently selected project. The MSI file is located and reloaded into the Software Warehouse repository if the date on the MSI file is newer than the last time it was loaded.

Update (force)

Select this menu option to update the currently selected project. The MSI file is located and reloaded into the Software Warehouse repository.

Update All Projects (if newer date)

Select this menu option to update all projects in the Software Warehouse. Each MSI file is located and reloaded into the Software Warehouse repository if the date on the MSI file is newer than the last time it was loaded.

Update All Projects (force)

Select this menu option to update all projects in the Software Warehouse. Each MSI file is located and reloaded into the Software Warehouse repository.

Database

Empty Database

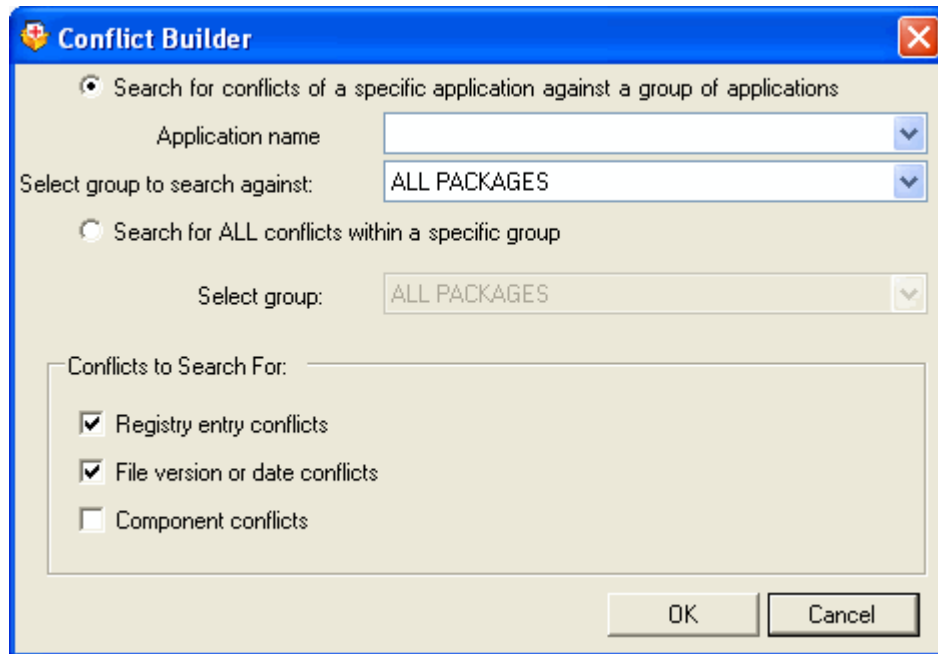
Select this menu item to remove all Projects, Applications and Groups from the database being used in the Software Warehouse. This option cannot be reversed. Click **Yes** to continue the removal. Click **No** to cancel this action.

Query Conflicts

*The Software Warehouse is only available in the Professional Version of MSI Studio.

Once the packages are imported into the Software Warehouse, MSI Studio can detect conflicts between packages and or packages within groups. Right-click in the Application/Project pane and select Query Conflicts from the popup menu. Query Conflicts can also be selected from the Warehouse menu.

The goal of the Conflict Builder is to reduce failed deployments by allowing administrators to find potential conflicts between packages before those packages are released. Packages can fail or programs can work erratically, when two applications install conflicting versions of a shared module. Conflict detection allows the repackager to see a potential conflict in his application when compared to the universe (or a subset) of the applications installed in their organization. This information is provided during package creation, or independently, and gives the repackager time to correct the problem before the package is deployed.



Search for conflicts of a specific application against a group of applications

Application Name

Enter the project name of the package to be compared against for conflicts.

Select group to search against

Select a group from the drop list. The selected group is compared against the specified application for conflicts. ALL PACKAGES may also be selected from the drop list. This will compare the specified application against all packages in the Software Warehouse.

Search for ALL conflicts within a specific group

Select group

Select a group from the drop list. All projects within the selected group are compared against each other for conflicts. ALL PACKAGES may also be selected from the drop list. This will compare all packages in the Software Warehouse for conflicts.

Conflicts to Search for

Registry entry conflicts

Select this box to include a query on registry entry conflicts.

File version or date conflicts

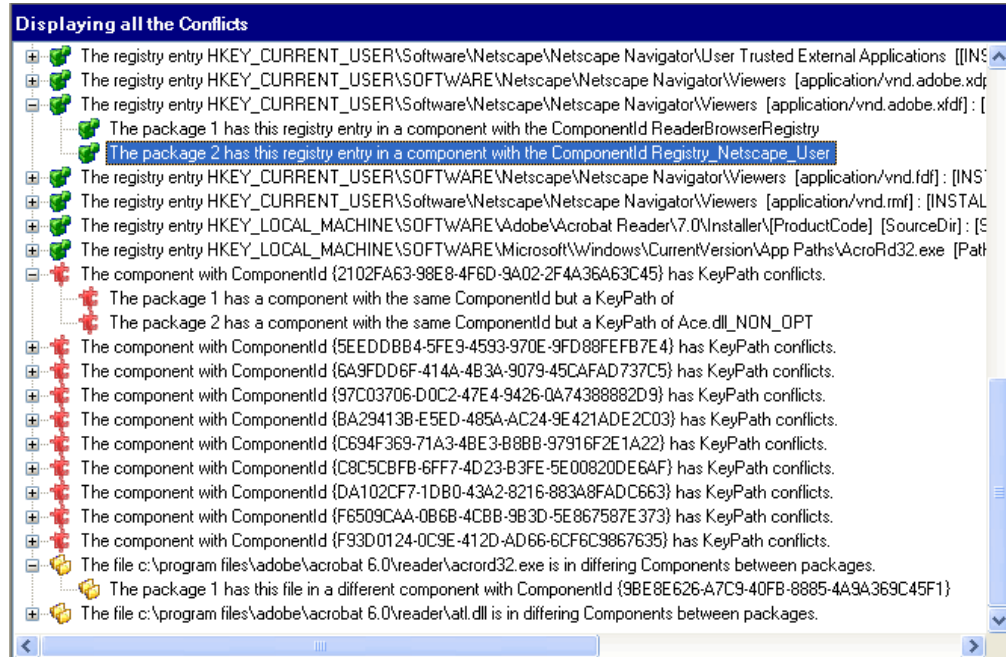
Select this box to include a query on file version and date conflicts.

Component conflicts




Select this box to include a query on component conflicts.

Conflict Builder Results

The results of the Conflict Builder are displayed in the Details (top left) pane of the Software Warehouse.



Conflict Builder Legend

-  Registry Conflicts
-  Component Conflicts
-  File Conflicts

Reference

COMMAND LINE OPTIONS

Option	Parameter(s)	Description
-B	Full Path to Package (i.e. "c:\my projects\thisproject.msi")	Initiates a command line build of the specified project. Be sure that the project does not contain any errors.
	Full Path to Package (i.e. "c:\my projects\thisproject.msi")	If there is no option specified, but the path to a valid existing project is, MSI Studio is launched and the project opened.
/diff	file1=xxx file2=yyy (where xxx is the path and filename of the first file to compare and yyy is the path and filename of the second file to compare.	Initiates the tool to compare 2 MSI files.

GLOSSARY

A

Application: Contains one or more related projects. For example, there may be several projects for a single application, each project is for a different version of the program.

G

Group: A Group holds a selection of applications that are related in some way. For example the Accounting Group may hold all applications that are common to the Accounting department.

M

MSI: Windows Installer Package containing all files, settings and configuration information for an application

MSM: Windows Installer Merge Module

MSP: Windows Installer Patch Package that is used to deliver updates to installed applications.

MST: Windows Installer Transform file contains a list of modifications to be made to the MSI file during installation of the application. Transforms can be used to provide information to dialogs where user supplied information is required.

P

Project: Contains information for MSI packages that are different versions of the same program.

T

Transform: see MST

W

Windows Installer: Microsoft's installation engine used to create MSI files for installation of applications

INDEX

A

Add Existing Project to Application
121

Add New Application 121

C

Command Line options 131

D

Delete Application 121