

InstallAware 6 Reviewer's Guide

*The Scripting Aware, Web Aware, MSI Aware
Installation Solution*

July 2006



The information contained in this document represents the current view of InstallAware Software Corporation on the issues discussed as of the date of publication. Because InstallAware must respond to changing market conditions, it should not be interpreted to be a commitment on the part of InstallAware, and InstallAware cannot guarantee the accuracy of any information presented after the date of publication.

This guide is for informational purposes only. INSTALLAWARE MAKES NO WARRANTIES, EXPRESS OR IMPLIED, AS TO THE INFORMATION IN THIS DOCUMENT.

InstallAware may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from InstallAware, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2006 InstallAware Software Corporation. All rights reserved.

InstallAware, MSIcode, Genuine Scripting for Windows Installer, WebAware Installations, Web Media Blocks, Partial Web Deploy, and One-Click Patching are either registered trademarks or trademarks of InstallAware Software Corporation in the United States and/or other countries.

Microsoft, Windows, and other Microsoft product names are trademarks or registered trademarks of Microsoft Corporation in the U.S. and other countries. All other marks are the property of their respective owners.

Contents

Overview	6
InstallAware: Advancing the Art of Setup Authoring	7
The Integrated Development Environment.....	11
One IDE, Multiple Views	12
The Visual Views.....	12
The Application Runtimes Visual View	13
The Update Packs Visual View	15
The Versions Visual View	16
The MSIcode View	17
In-Place Editing	18
Visual Command Editors.....	18
Code Folding.....	19
Multiple Editor Windows	19
Code Navigation	20
Round-tripping IDE	21
Dialog Designer	21
Object Properties Window.....	22
Setting Object Rules	23
Setting Object Behaviors	24
Rich Collection of Tools.....	25
Setup Decompressor	25
Group Policy Wizard	27
Localization Wizard.....	28

Form Batch Update Tool	29
Code Signing Tool	30
Database Validator	31
Desktop Layouts	32
Watches Window	32
Project Manager Window	33
InstallAware 6 Wizards, Templates, Converters, Samples	34
Default Project	35
Project Wizard	35
PackageAware	36
Database Import Wizard	37
Templates	38
Plug-Ins	38
Converters	39
Samples	40
Refactoring	42
IDE Options	43
Project Options	44
MSIcode.....	46
Types of MSIcode Commands	46
Achieving Conditional Program Flow with Windows Installer.....	48
A Single Script for Installs, Maintenance, and Uninstalls	49
Limitations of MSIcode.....	50
“Non-Windows Installer” MSIcode Commands.....	51
Get System Information	52

Advanced Install Actions	52
Run Programs, Call Dynamic Link Libraries.....	52
Plug-In Extensibility	53
Two-Way Integrated Visual View	54
Automation Interface	54
Compiler Variables	55
Integrated Debugging	57
WebAware Installations (Partial Web Deployment)	60
Web Media Blocks	60
Superior Compression	63
Repacking Databases	65
One-Click Patching.....	66
Windows Installer Functionality	70
Logo Certifiable Setups	70
Advertising	71
Install-on-Demand	71
Self-Healing Installations.....	72
Elevated Privileges	73
Corporate Deployment.....	74
Shelling to Other MSI Installations	75
Summary	77
About InstallAware Software Corporation.....	77

Overview

InstallAware[©]
6



Overview

Welcome to the InstallAware 6 Reviewer's Guide. This document will familiarize you with InstallAware 6, the newest version of InstallAware Software Corporation's flagship setup development environment, culminating more than ten years of technological innovation.

The InstallAware 6 Reviewer's Guide is organized into two parts. In this first part, the Overview, you will find a general introduction to InstallAware 6.

The second part of this guide takes you on a tour of InstallAware 6. It is organized by the major areas of installation development and support in InstallAware 6, providing you with an overview of each area, and is followed by a description of the many updates, enhancements, and additions introduced in this release. If you are already an InstallAware enthusiast, you may want to quickly scan the overview section, concentrating instead on the updates that make this the most important version of InstallAware yet.

InstallAware: Advancing the Art of Setup Authoring

InstallAware's legacy began with a simple vision: to make setup authoring simple and enjoyable again. Since the debut of Windows Installer in 1999, setup authoring has become an incredibly complex endeavor. Developers are forced to work within the constraints of a relational database structure when building their setups, and they can no longer use branching script flow to direct their setup logic. Even creating a very simple setup that installs a single file entails populating multiple interconnected database tables. For more sophisticated installations, custom actions often have to be utilized – a labor intensive and error prone process. While a large variety of third party setup authoring tools have emerged since 1999, automating or visualizing most of the tedious work, Windows Installer still remains a setup development platform that is very hard to develop for and customize.

InstallAware has been designed from the ground up to solve today's setup authoring challenges. InstallAware targets the Windows Installer platform, because despite its being hard to use, Windows Installer has become the corporate standard worldwide for developing and deploying application installations. Most companies today will not even consider software that is not made available with a Windows Installer setup package. With InstallAware, you have everything you need to create modern setups that address today's development needs:

- **MSIcode:** In addition to the visual setup editors found in most setup authoring tools, InstallAware provides a human readable, branching setup script called MSIcode. Available only with InstallAware, MSIcode offers a drag-drop scripting environment that introduces branching, script-driven execution logic into Windows Installer. Yet, InstallAware does this without requiring any external scripting engine that needs to be installed as a scripting runtime before-hand. InstallAware automatically compiles the MSIcode setup script into a pure, independent, 100% logo compliant MSI file at build time. This shifts the burden of properly populating MSI database tables and creating custom actions (the latter an especially error prone process) from the developer to the setup IDE, providing developers with unprecedented levels of flexibility in coding their setup logic. Such flexibility was unavailable since the initial launch of Windows Installer in 1999.
- **WebAware Installations:** InstallAware allows splitting the data contained inside a setup package into multiple online and offline parts. Web deployment traditionally involves putting the entire setup online, including the vital files that are always installed. This mandates an active web connection to install any part of the product. InstallAware sidesteps this artificial limitation and allows developers to precisely choose which parts of their setup are to be downloaded, and which are to be provided with the main setup program. Isolating runtimes and rarely used application features from the main setup download saves time and bandwidth for end users installing the applications, and also for companies hosting downloads. Application runtimes which may already be present on end user systems, and typically bloat setup downloads, are ideal candidates to benefit from WebAware Installations. With WebAware Installations, they will only be downloaded if required, and the main setup is wholly self-contained, capable of executing an installation without requiring an active Internet connection.
- **Full Support for all Windows Installer Standards:** Every setup that is generated by InstallAware is in full compliance with all Windows Installer standards and provides uncompromised support for all Windows Installer features. If you are already familiar with Windows Installer, you will recognize such features as install-on-demand, self-healing setups, advertising, binary differential patching, merge modules, silent/command-line installations, and so on. InstallAware fully supports each of these Windows Installer features while continuing to extend Windows Installer to add its own.
- **Other Unique Benefits:** InstallAware combines the above core technologies to deliver compelling installers in record time. InstallAware setups can authenticate users online, define setup features on-the-fly, check for application updates on the web, install multiple instances of the same product, check for serial numbers, build installers for multiple editions of a product from a single installation project, install drivers, install complex application runtimes like Microsoft SQL Server, run SQL scripts, configure IIS websites/folders/application pools, and do much more.

What Developers Say About InstallAware:

I'm continually impressed by the depth of power that InstallAware offers me. I know absolutely nothing about creating Windows MSI based installers, but with InstallAware I don't need to. It's easy to quickly get up and running with InstallAware, but there is so much more to explore once you start getting more adventurous with your MSIcode scripts. And the support for custom dialogs is truly awesome! *David Clegg, eBetOnline*

InstallAware has enabled me to fully support Windows Installer, yet remain in full control of what's going on. InstallAware's WebAware installations help to ensure that my customers only download what's necessary, and the excellent compression ensures that it's small as possible. I take quality very seriously, and I've been impressed with what I've seen. *Troy Wolbrink, TntWare*

InstallAware is a one-of-a-kind product. I searched high and low for a solution to provide the ability to download and install runtimes from the Internet only if they are needed. InstallAware was the only program I could find that offered this for all the runtimes needed. Add to that the dialog editing/customization, and all the other great features; InstallAware is the best solution available to my knowledge. *Chris Hoffman, Mystik Media*

A first class Installer using the Microsoft Windows Installer Engine. You are not limited to standard MSI actions; you can even author your own plug-ins. Combined with an extensive selection of pre-built themed installers, or even the option to totally construct your own, InstallAware becomes one of the most flexible packages out in the market today. *Andrew Neillans, ABCC Computers*

InstallAware is the best MSI based installer available at this time. It will quickly handle small jobs but will grow to meet the more complex requirements of larger jobs. As software developers we really like the ability to drop down to the InstallAware scripting level which is still quite easy to use yet flexible enough for our complex tasks. Most MSI based installers that we looked at were difficult to use because of a lack of ability to drop down to a scripting style model and were expensive to boot. Our applications that are released today and in the future are dependent upon new Microsoft technologies such as the .NET framework, MDAC and many others. It's nice to have the benefits of an MSI based installer to make sure these pieces are installed while still providing enough power and flexibility to complete the main job of installing our product. *Jim Gunkel, Nevrona Systems*

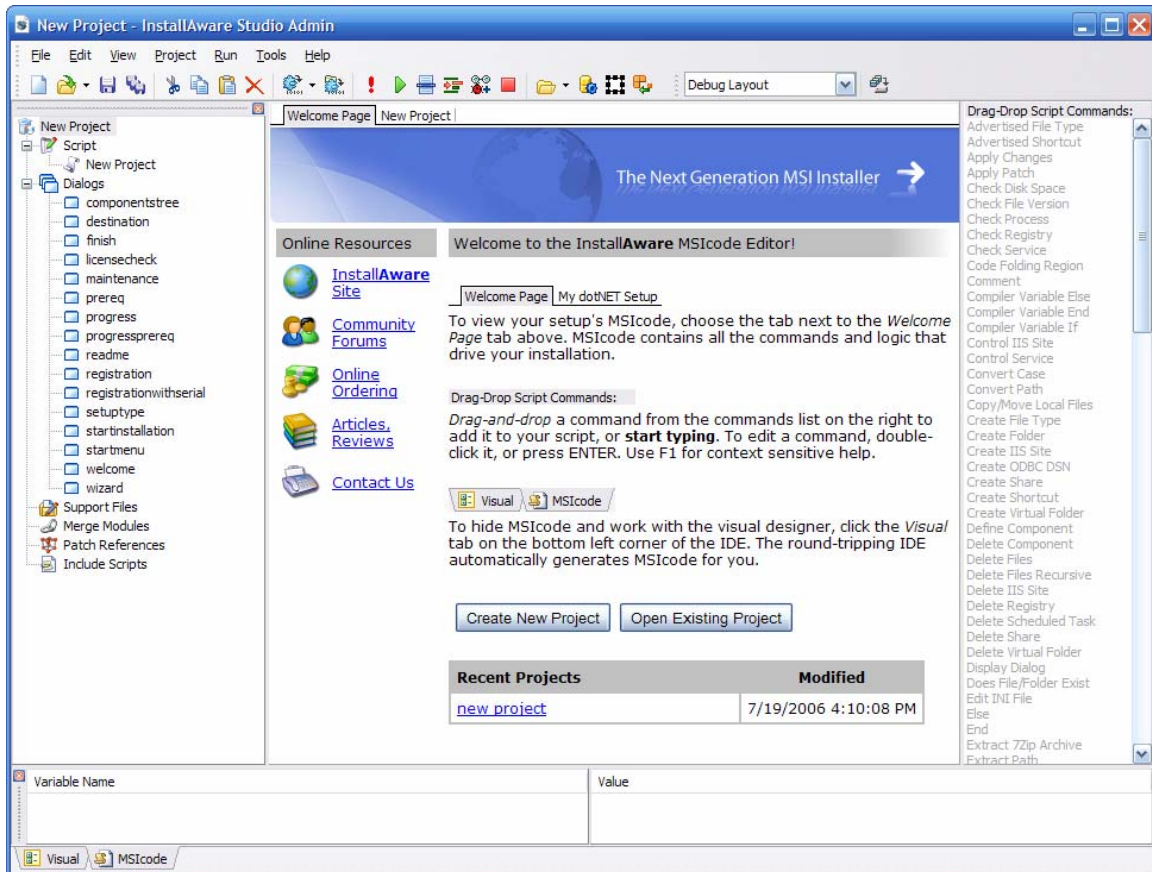
The InstallAware 6 IDE

InstallAware[©]
6



The Integrated Development Environment

The InstallAware 6 IDE (integrated development environment) represents state-of-the-art in setup authoring tools. Fast, clean, and visually appealing, InstallAware 6's IDE enables you to develop setups faster and better.

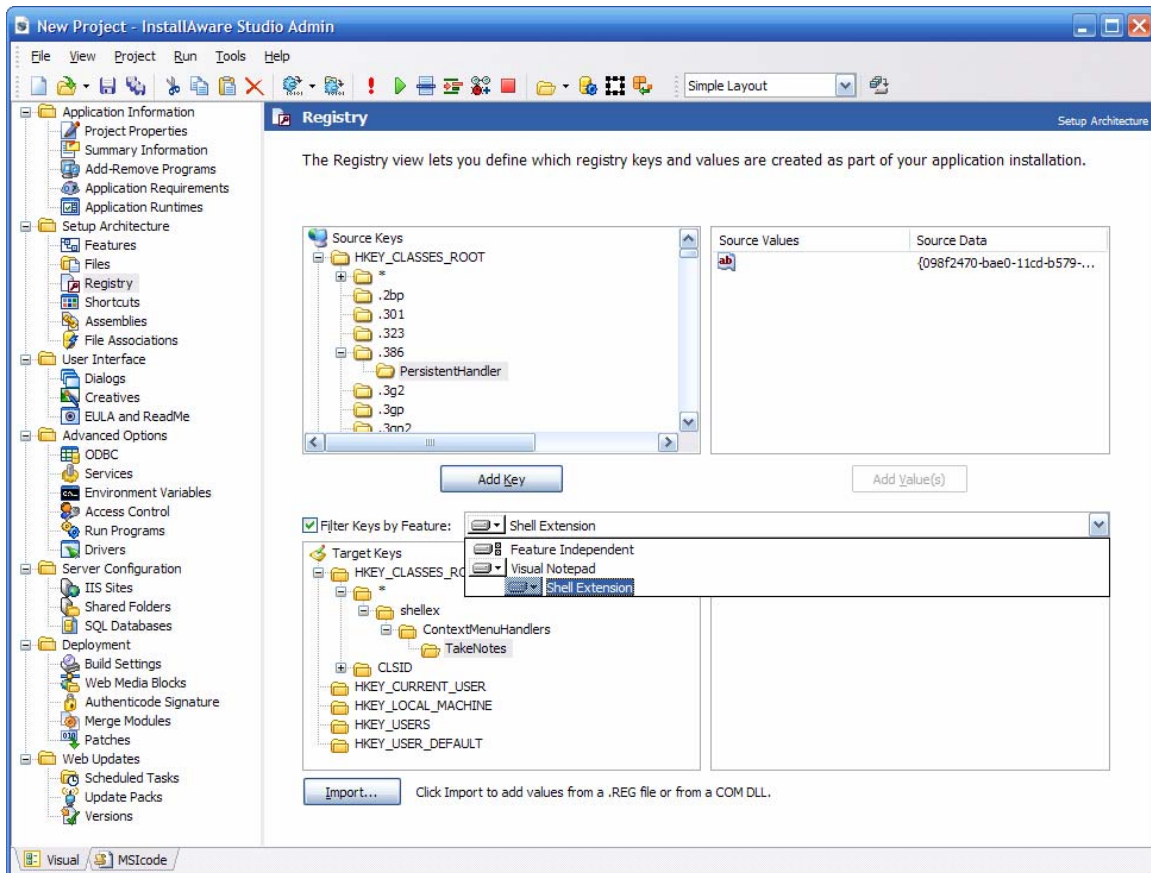


This section focuses on the features found in the various panels, designers, dialog boxes, and views of the IDE.

One IDE, Multiple Views

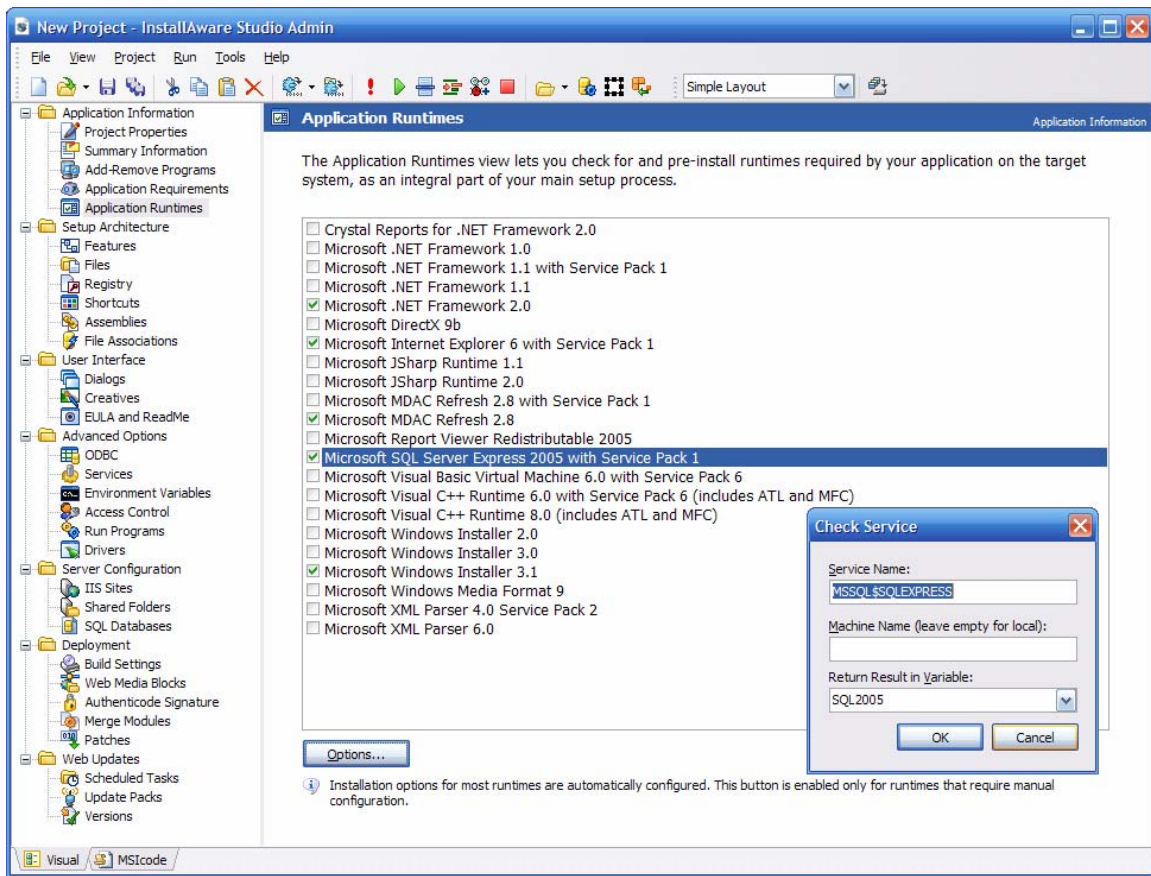
Whether you are coding a complex setup with highly customized scripting logic, or a simple one that does not require any tweaking, InstallAware 6's IDE provides you with a consistent and powerful set of views designed to increase your productivity.

The Visual Views



The first of the two main IDE views is the visual view. The visual view provides a purely visual representation of your underlying MSIcode script. As you make changes visually in the visual view, the IDE will automatically update the underlying MSIcode for you – transparently and seamlessly, without you writing a single line of code. The many pages available on the visual view enable rapid setup development and eliminate the need to manually edit your MSIcode scripts in most cases.

The Application Runtimes Visual View



InstallAware 6 provides a time saving Application Runtimes visual view. This view, which is also extensible with custom runtime installations built by setup developers, enables single-click deployment of complex application runtimes, together with their nested dependencies, and provides a consistent development and end-user installation experience when working with a diverse array of application runtimes. Application runtimes are installed through a “setup pre-requisites” dialog, as an integral part of your main installation, and no separate, potentially confusing installer interfaces are shown at any time. Errors or missing system updates, such as critical service packs, are handled correctly with helpful messages shown, enabling your end-users to resolve update issues on their own without having to consume your technical support resources.

For instance, when the Microsoft .NET Framework 2.0 runtime is checked in this view, the IDE automatically includes all dependent runtimes: Windows Installer 3.0, and Internet Explorer 6 with Service Pack 1. The installation of each runtime is very complex and varies by platform. Windows Installer 3.0 is

not supported on Windows 95, 98, and ME operating systems, nor is it required when installing Microsoft .NET Framework 2.0 on those platforms. The nested Windows Installer 3.0 installer already provides the complex logic necessary to install this runtime on all supported platforms, which is leveraged in turn by the Microsoft .NET Framework 2.0 runtime, with the click of a single check-box in this new visual view.

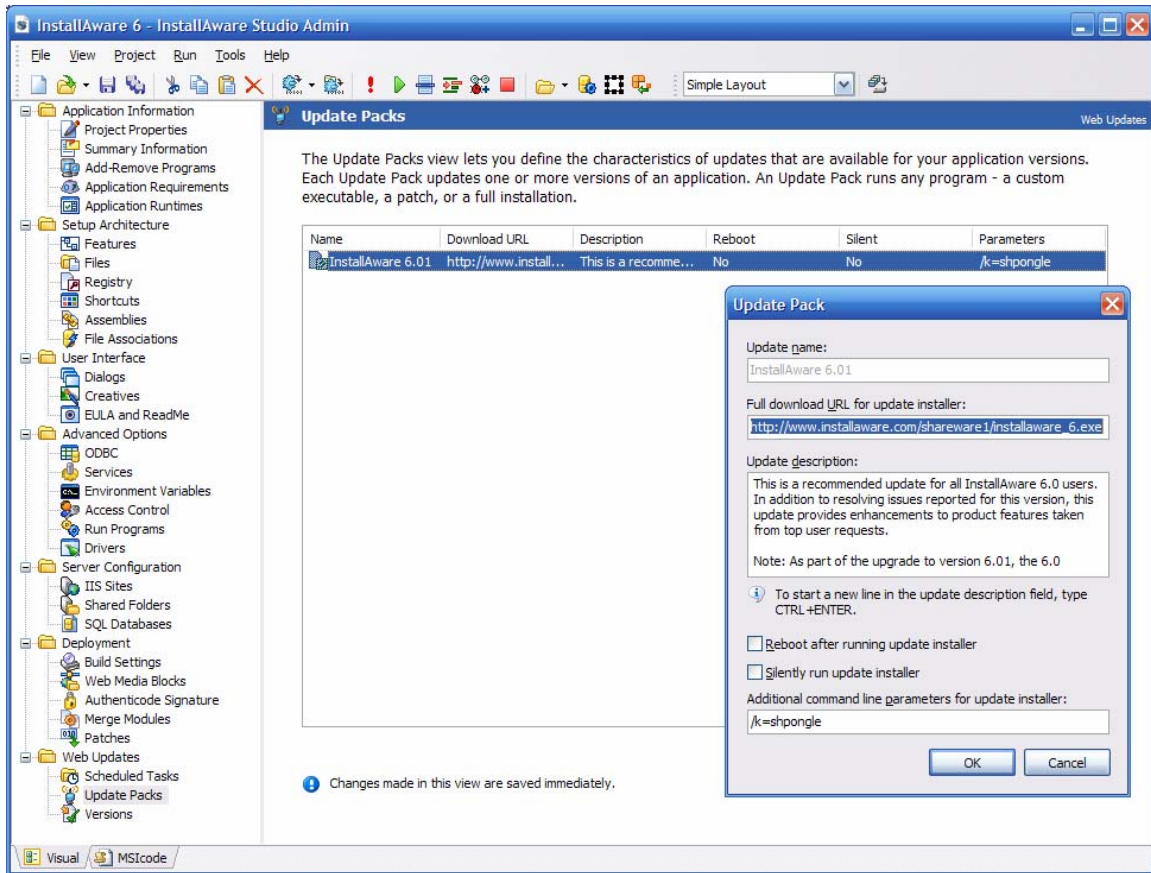
Clicking the Options button displays helpful configuration options for the runtime being installed. As another example, the Options button used on the Microsoft SQL Express 2005 runtime allows customizing the instance of the server being installed, so the runtime is installed only if the desired instance is not found. All this happens without ever having to use the MSIcode view, which of course remains available if further customizations are required.

By deploying your application runtimes with InstallAware, you will also experience an up to 66% reduction on the size of your built installation packages. This is made possible with InstallAware's superior compression levels and is available at no effort or cost to you – just deploy your application runtimes with InstallAware.

The Application Runtimes visual view offers a one-stop location to painlessly install, customize, better compress, and intelligently handle error conditions that may occur during installation – all at the click of a single check-box. Custom runtime installers that are plugged into the InstallAware 6 IDE leverage all the pre-built logic for installing their dependent runtimes, at no effort to the setup developer. Of course, custom runtime installers can also use the “Options” button to permit setting options on the custom runtime, again without having to consult the MSIcode view.

Thus, the Application Runtimes view in InstallAware 6 provides the ultimate force-multiplier for developers building application setups, and saves tremendous amounts of time and money both during and after development: end-users spend less time downloading, consume significantly reduced technical support resources, and your bandwidth costs are cut by up to 66%, especially considering the bulky sizes of the runtimes that are supported in this view.

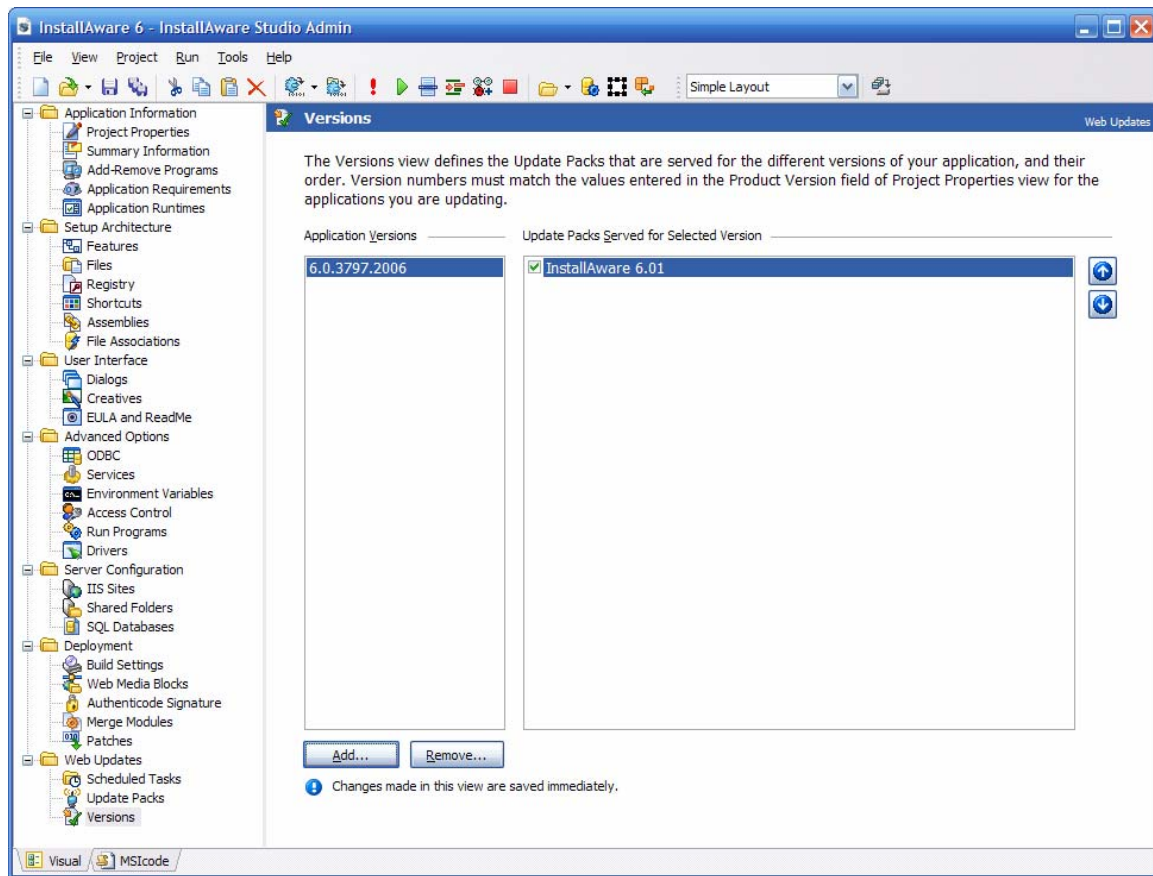
The Update Packs Visual View



InstallAware 6 introduces Web Update capability for your installers. Unlike other expensive web update solutions, InstallAware's Web Update mechanism is free, completely customizable, and ships with full source code, accessible from the MSICode view.

The Update Packs view lets you define the modular blocks that update your applications and their components. Each Update Pack consists of an executable program that is downloaded and run. The executable program may be a patch (built using InstallAware 6's incremental patching option), a full blown new version installer (with the capability of automatically uninstalling the old version), or any other custom program – you do not have to ship an installer or a patch as your update, but can provide any custom program which fulfills your needs.

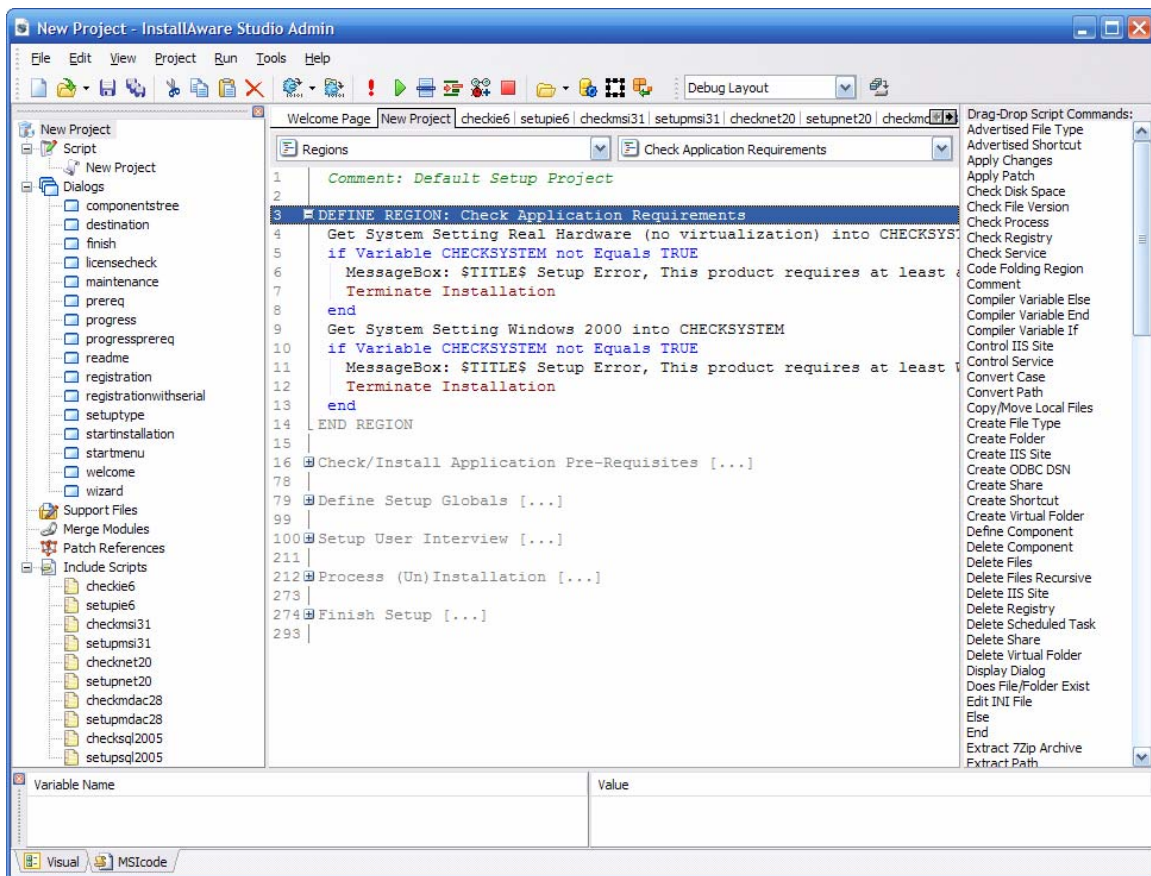
The Versions Visual View



Once update packs have been defined in the Update Packs view, they are assigned to applications already deployed on the field using the new Versions view. This view associates one or more update packs with particular versions of an application. The InstallAware 6 Web Update mechanism will download each selected update pack and run it on the end-user system, bringing your end-users fully up to date with your latest product enhancements. Each update pack may be freely assigned to one or more application versions, providing flexibility in updating your applications.

Another new view in InstallAware 6 is the Scheduled Tasks view, which lets you schedule tasks on end-user systems using the built-in Windows Task Scheduler. The Web Update mechanism internally uses scheduled tasks to check for updates on a regular basis. You may leverage the same functionality to schedule your own custom tasks.

The MSIcode View



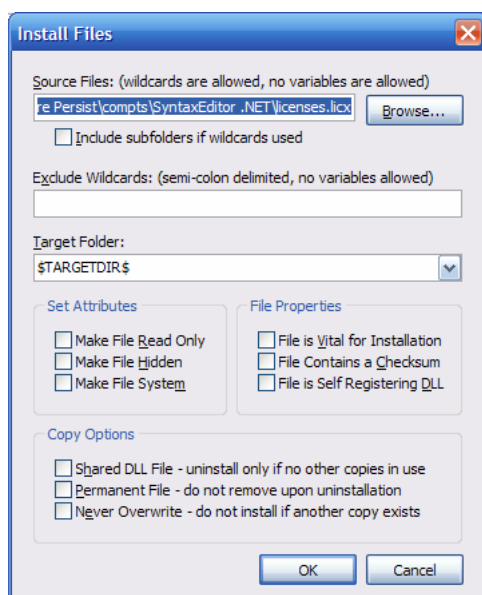
The MSIcode view offers unfettered access to the heart of your installation. One of the strengths of InstallAware is that nothing is hard-coded in the setups it creates. You have complete control over your setup program, and the MSIcode view is the place to exercise that control. What's better, you do not need to memorize a new scripting language, its syntax, or command set either! Simply drag-drop commands from the command list on the left to the MSIcode script on the right. Your commands will be inserted into the MSIcode script at their drop point.

In-Place Editing



With InstallAware 6, it is also possible to directly type text into the script editor. Matching commands are automatically highlighted in a combo-box that automatically appears. Pressing ENTER selects the closest matching command to the letters typed so far, or the combo box may be scrolled for a list of commands that are near by alphabetical ordering. Being able to directly edit text in the script editor dramatically accelerates the authoring process for the MSIcode script.

Visual Command Editors



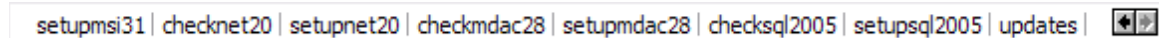
To edit the parameters of a command, just double-click it. A dialog box pops up with the switches that are available for that command, and if you are unsure – help is just one key press away. Press F1 at any time for context sensitive help anywhere in the InstallAware 6 IDE. Of course, the MSIcode view offers all the conveniences one is used to find in traditional source code editors such as cut/copy/paste/delete, find, commenting in/out blocks of code, moving around blocks of code, and so on.

Code Folding

```
1  Comment: Script to Schedule, Download, and Install Updates
2
3  [compiler if Variable BUILDMODE not Equals PATCH]
4
5  Update Mode? [...]
58
59  Schedule Updates [...]
145
146  Download and Install Updates [...]
346
347  label: End of Updates
348  [compiler end]
349
```

Another feature in InstallAware 6 is the ability to fold pieces of code. Code folding has become a standard part of text editors in modern IDEs, and InstallAware follows suit. Folding blocks of code not only makes the setup script easier to navigate, it also provides for a clearer organization of the different regions of the script.

Multiple Editor Windows



In tandem with the Include Script command in InstallAware 6, the IDE displays multiple editor tabs to enable switching between different setup scripts. This makes managing complex setup projects a lot easier, and also powers team development efforts.

A welcome page is also available; listing frequently used online resources, product literature, and recently used projects in one convenient location.

Code Navigation

```

233
234 Comment: Modify Target System
235 Perform Uninstallation [...]
243 DEFINE REGION: Perform First Time or Maintenance Installation
244 Comment: Install/Re-Install product
245 Create Shortcut $SHORTCUTFOLDER$\Uninstall New Project to $UNINSTALL
246 [OFFLINE CONTENT]
247 [DEFINE WEB MEDIA Shell Extension]
248 Get Component Visual Notepad\Shell Extension Selection State into Variable SELECTED
249 if Variable SELECTED Equals TRUE
250     Write Registry Key HKCR\CLSID\{07A06366-4BE7-419C-8F25-7996C...}
251     Write Registry Key HKCR\CLSID\{07A06366-4BE7-419C-8F25-7996C...}
252     Write Registry Key HKCR\CLSID\{07A06366-4BE7-419C-8F25-7996C...}
253     Write Registry Key HKCR*\shell\ContextMenuHandlers\TakeNotes
254     Write Registry Key HKCR*\shell\ContextMenuHandlers\
255     Write Registry Key HKCR*\shell\
256     Install Files C:\Program Files\Take Notes\mTakeNotes.dll to $TARGETDIR$
257 end
258 [DEFINE WEB MEDIA Visual Notepad]
259 Get Component Visual Notepad Selection State into Variable SELECTED
260 if Variable SELECTED Equals TRUE
261     Install Files C:\WINDOWS\notepad.exe to $TARGETDIR$
262 end
263 Comment: TO-DO: Insert any additional install commands here
264 [compiler if Variable BUILDMODE Equals PATCH]
265     Apply Patch (get result into variable SUCCESS)
266 [compiler else]
267     if Variable ADVERTISE Equals TRUE
268         Apply Advertised (get result into variable SUCCESS)
269     else
270         Apply Install (get result into variable SUCCESS)
271     end

```

Code navigation drop-downs in the InstallAware 6 IDE make finding vital parts of the setup script a no-brainer. Any component, web media block, comment, label, code folding region, or setup execution block can be located instantly using the drop-downs.

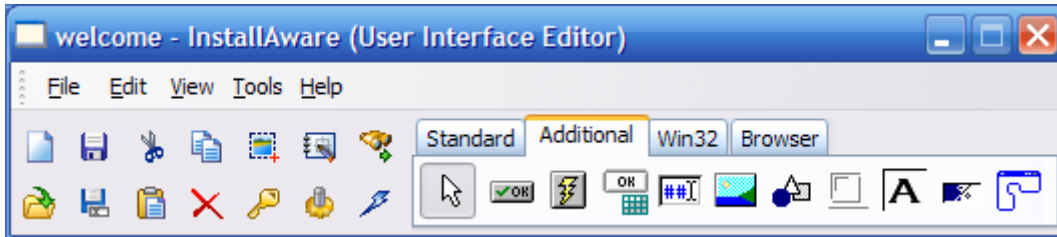
Round-tripping IDE



As you work in the MSICode view, whenever you switch back to the visual view, the visual view will properly render the latest changes you have made to your script. The visual view does a great job of keeping up with your modifications even in highly custom scripts – bringing you true two-way integrated setup editing that saves you time and effort. All you need to switch views is just click the dog ears on the bottom of the IDE window.

Dialog Designer

The InstallAware 6 IDE ships with a state-of-the-art dialog designer. Using the dialog designer you can completely customize all aspects of your setup windows, and even design new ones from scratch. InstallAware 6 ships with a pre-built collection of fifteen setup themes which you can use out-of-the-box, or as a starting point for own customizations. You can of course create and publish your own themes as well.



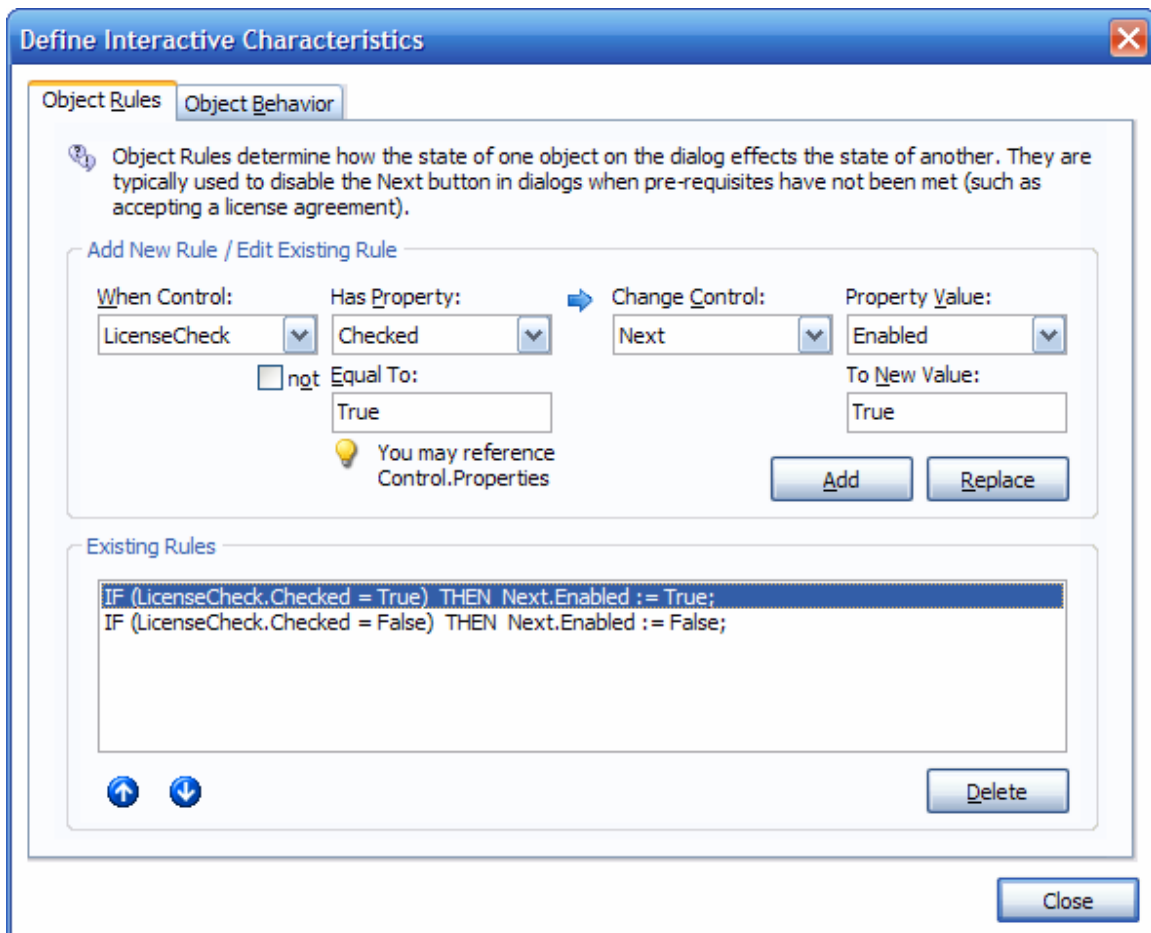
The component palette on the dialog designer will seem especially familiar to developers who have used visual programming languages such as Visual Basic, C#, C++ Builder and Delphi. Simply select a control on one of the palette pages and then click on the window being designed to insert an instance of that control. The dialog designer offers a very wide variety of dialog controls, including unique items that are not available with any other setup development environment: Flash containers for displaying interactive Flash tutorials/movies/animations during installation progress, HTML containers for rendering sophisticated HTML content, visual browsers styled after native Explorer tree/list/combo controls for quick and easy selection of files and folders, and so on. InstallAware 6 adds marquee style progress bars, hollow/non-rectangular dialogs, custom bitmap buttons, and tabbed notebooks to the collection.

Object Properties Window



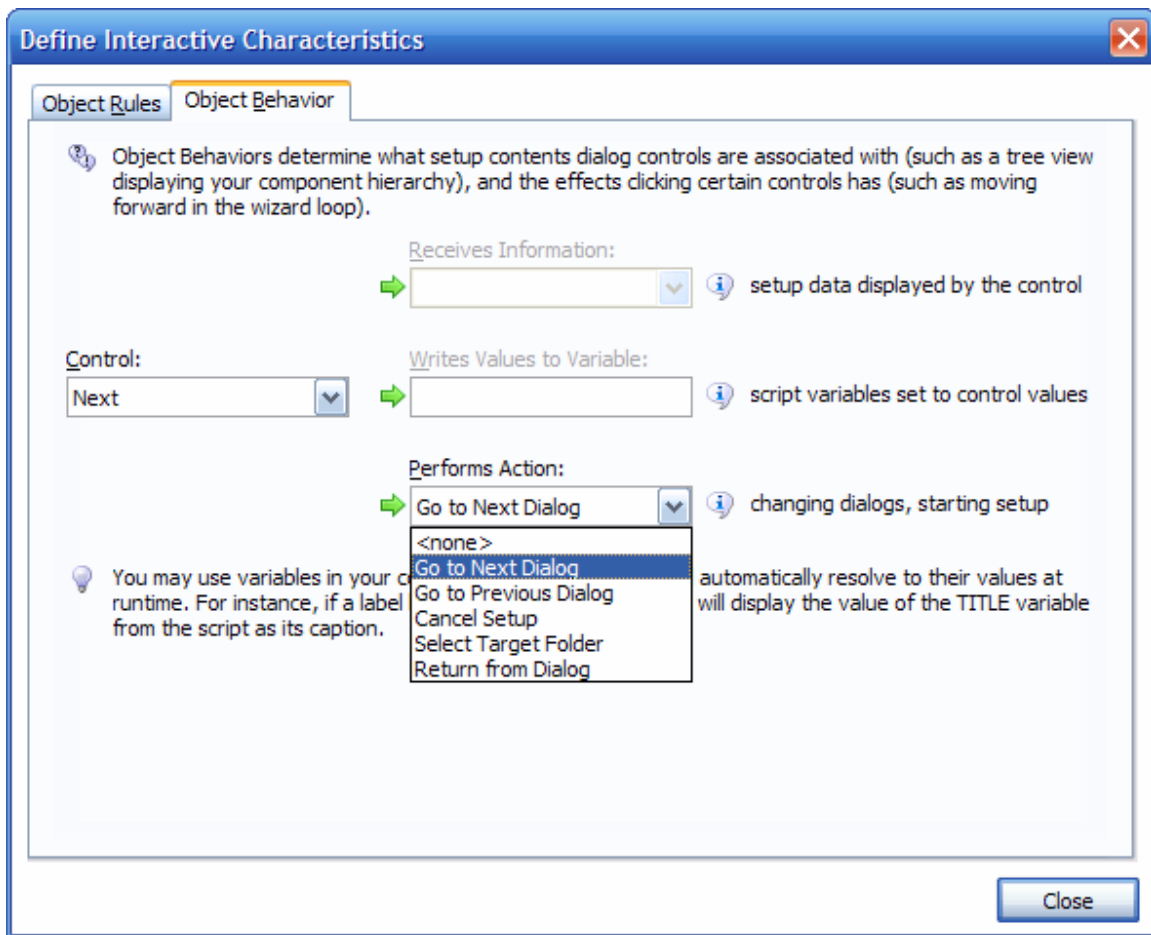
The Object Properties window lets you view and customize the many published properties for the controls that you place on your dialogs. Set text captions, load/save bitmaps, choose animations, set visibility, transparency, and do more in this window.

Setting Object Rules



Having controls on your windows interact with one another is very easy. Just double-click any control to bring up the Object Rules editor. This editor lets you visually configure how controls respond to state changes of other controls on the same window. For instance, if a license checkbox is unchecked, you can disable the Next button on the window, to prevent users who do not accept your license agreement from installing your product.

Setting Object Behaviors

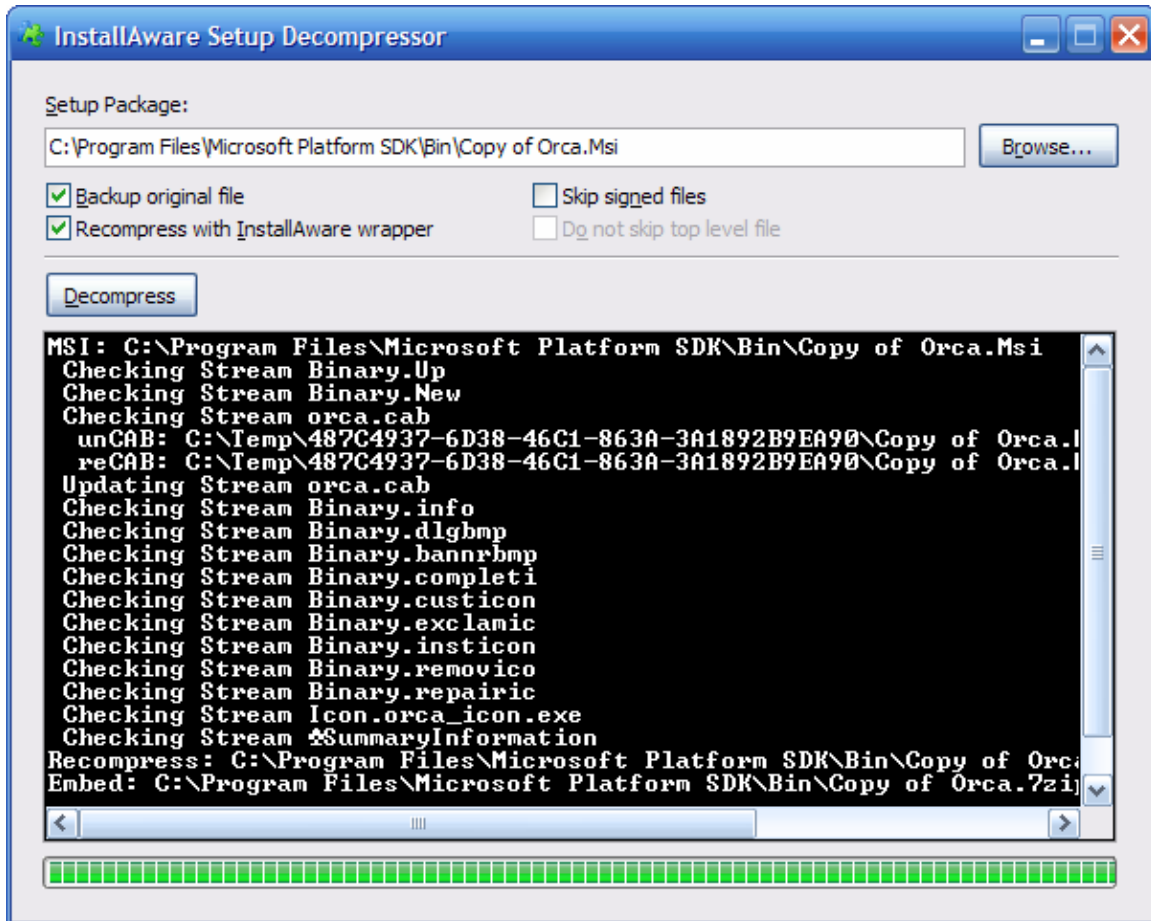


Passing values of variables between the MSIcode script and dialog controls is also very easy. Choose the Object Behavior tab after double-clicking any control, and simply type in the name of the script variable that should get/set the state of the control on the window. The Object Behavior tab also lets you configure controls for receiving pertinent setup information, such as progress percentage, textual description of the current installation task, and so on. Controls capable of initiating actions, such as moving to the next wizard dialog, are configured here as well.

Rich Collection of Tools

The InstallAware 6 IDE is complemented by a rich array of tools which assist with everyday setup authoring tasks, helping automate tedious chores and performing other useful functions.

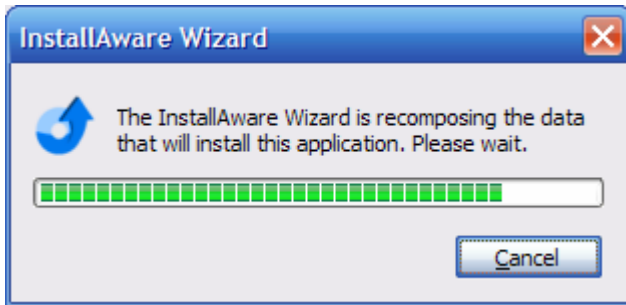
Setup Decompressor



The Setup Decompressor tool lets you increase the compressibility of external Windows Installer databases and various third party setup files that may be included with your setup package, such as merge modules (MSM files), installation databases (MSI files), and setup executables (EXE files). While InstallAware 6's compression works miracles in reducing runtime and application sizes, pre-compressed data cannot be recompressed, no matter how good the compression algorithm is. This tool processes setup files and

retrieves a list of compressed cabinet streams which are then temporarily extracted, repacked with zero compression, and then reinserted back into the original file – all with the actual data and files left intact.

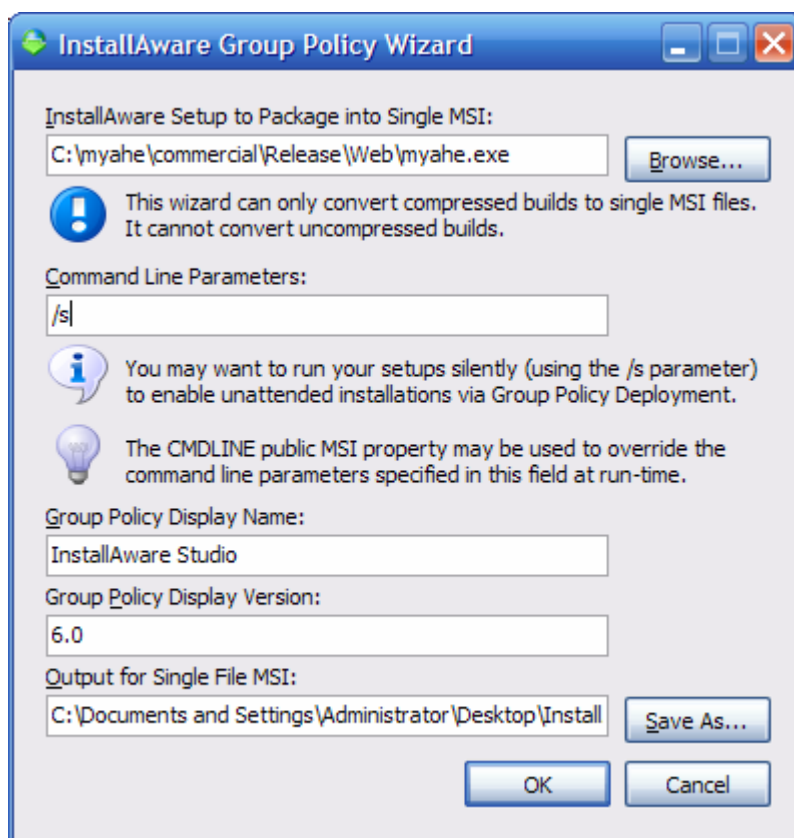
This procedure assures you will get the smallest possible file sizes when you add these setup files to a larger InstallAware installation, because now the decompressed data can be compressed into a much smaller size using InstallAware 6’s superior compression algorithm.



New in InstallAware 6 is the ability to create independent installation packages out of these decompressed setup files, without needing to enclose them inside a master InstallAware package. The “Recompress with InstallAware Wrapper” check-box will produce setups that can be run independently of an InstallAware installation, while still providing the superior compression rates made available with InstallAware.

Whether you wish to make your own installers as small as possible, or simply to trim off the fat from existing installers you may have already built in other third party tools, the Setup Decompressor tool works miracles in compressing data and eliminating bloat from installations. The application runtimes distributed with InstallAware have also been pre-processed using the Setup Decompressor, facilitating the best runtime compression rates in the industry.

Group Policy Wizard

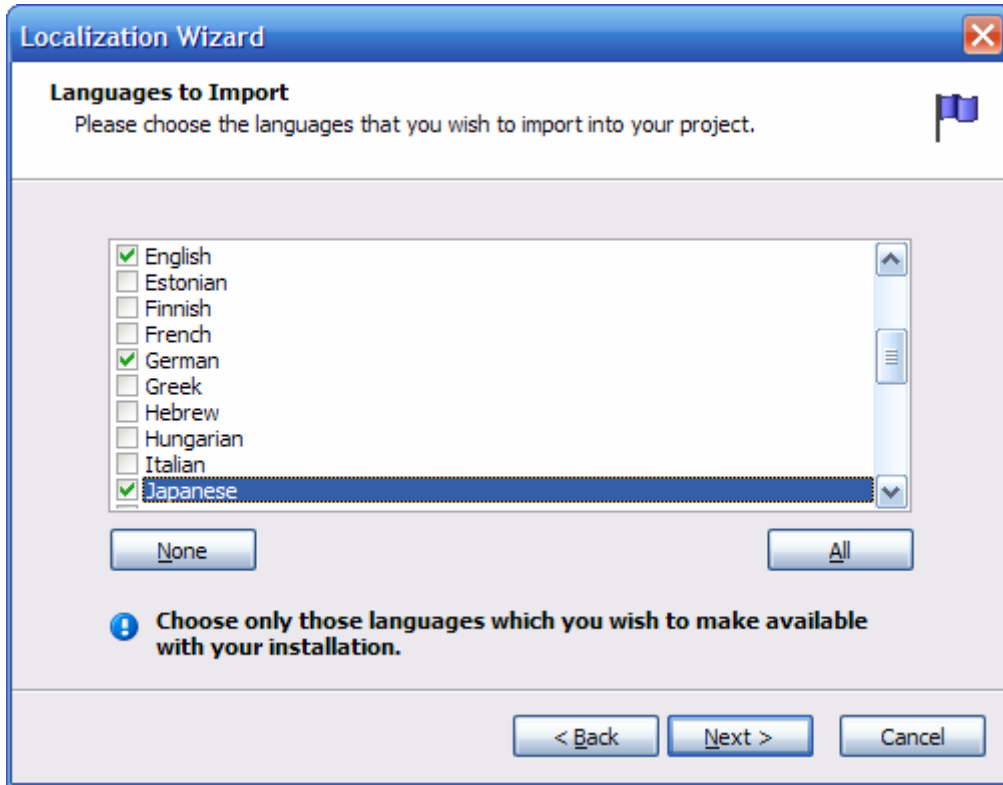


Another valuable tool in InstallAware 6 is the Group Policy Wizard. While InstallAware setups are based on MSI technology, and InstallAware produces pure MSI files which run the entire installation through the Windows Installer engine, some advanced features of InstallAware such as partial web deployment and superior compression are handled by an executable which downloads data and prepares the MSI for execution before invoking the Windows Installer engine. As executable files cannot be distributed using Active Directory deployment, which only accepts single MSI files as input, the Group Policy Wizard wraps InstallAware setups that may be comprised of multiple files inside a single, outer MSI file. This outer MSI is Group Policy friendly and works seamlessly with Active Directory deployment – automatically deploying, advertising, installing, and uninstalling the inner InstallAware setup on demand as configured by the system administrator.

The Group Policy Wizard also permits customizing the command line passed to the embedded InstallAware setup, either by hard-coding a command line in the wizard itself, or by setting the public

CMDLINE property of the outer MSI at deployment time on end-user systems. Therefore a single MSI file can be made available for use by both end-users and in corporate deployment scenarios using the Group Policy Wizard.

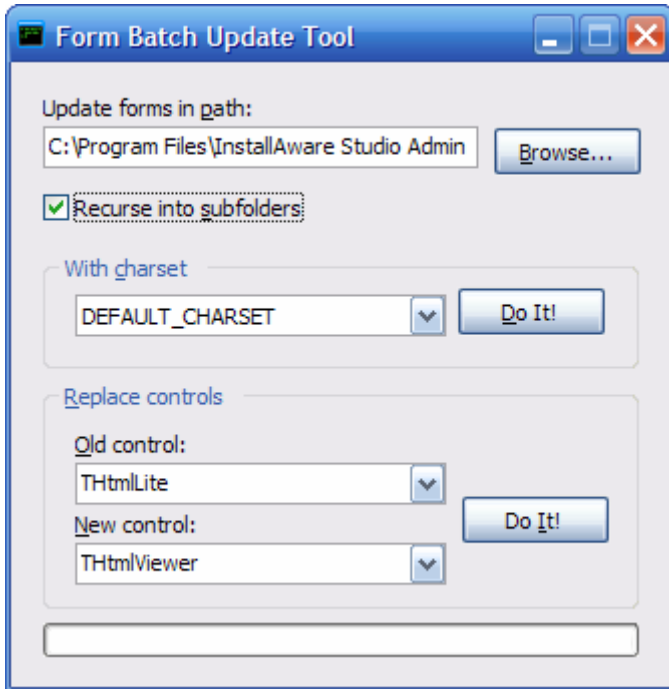
Localization Wizard



The InstallAware 6 Localization Wizard makes it a snap to extract, localize, and import strings used in a setup project. This wizard first scans a setup project, its MSIcode scripts, and all install wizard dialogs; making a list of all the localizable strings located throughout the project. It then stores this data inside a folder, adding a visual translation tool. The visual tool may then be used to localize the product into any of the supported 34 languages, either on-site or off-site. There are no distribution restrictions for the visual localization tool, so the entire contents of this folder may be sent to an off-site translation service, who are then able to localize the entire setup project without requiring access to either the main InstallAware IDE or the actual setup sources. Once the localization is complete, the main setup developer can then run the Localization Wizard again, and import the localized strings into the setup project, building multi-lingual setups capable of switching languages at either design time or run time. As further modifications are made

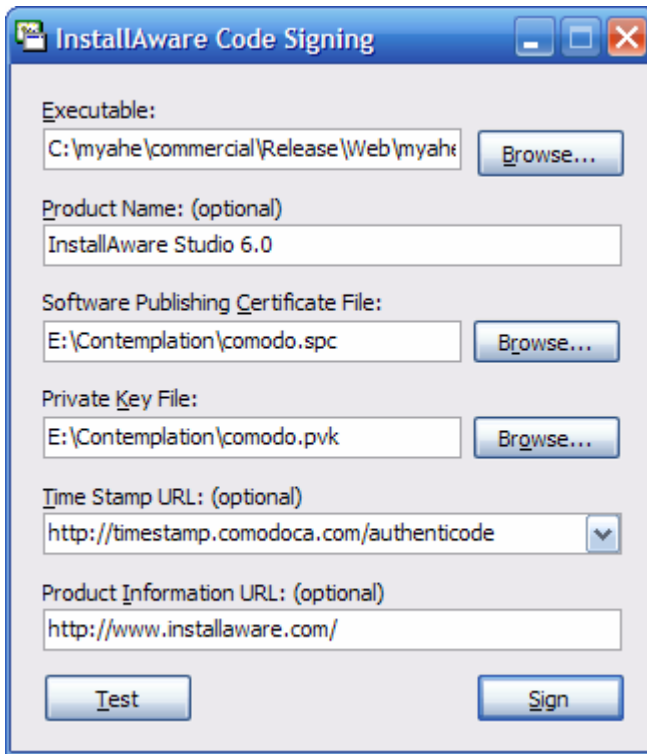
to the project, the Localization Wizard keeps track of what has changed and retains existing localization information, preventing the duplicate work of re-localizing strings that have been previously translated.

Form Batch Update Tool



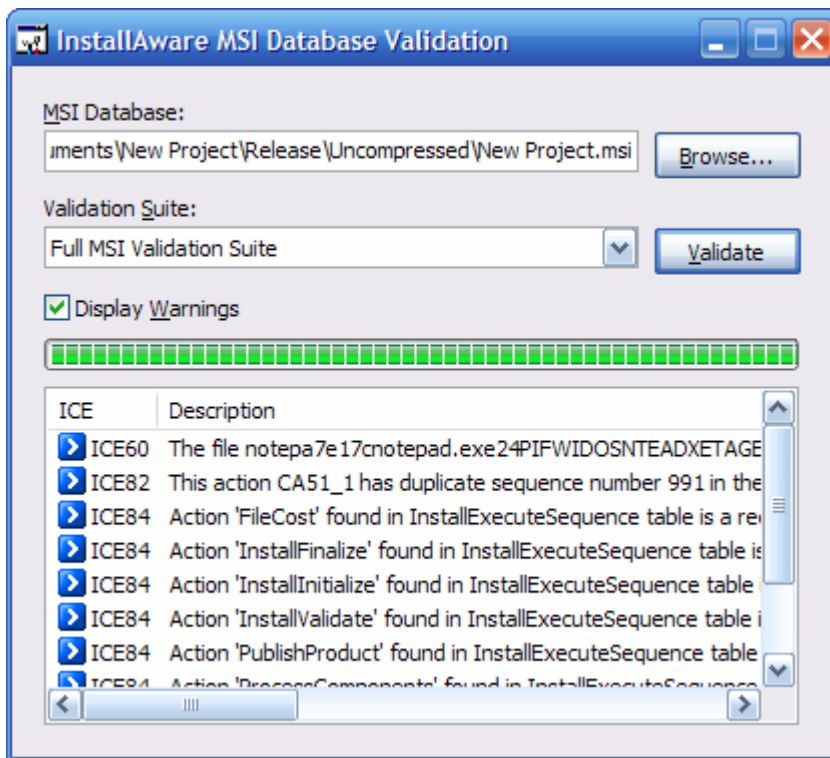
The Form Batch Update Tool updates multiple dialogs in a single pass. You may replace a particular kind of a control used on a dialog with another kind, or update the character set used for fonts on your dialogs.

Code Signing Tool



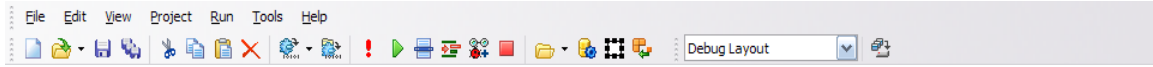
The stand alone code signing tool lets you sign your executables using Authenticode technology, and a software publishing certificate/key pair. Authenticode code signing validates and protects the integrity of your installation executables and provides information on the publisher. This information assures end users that the program they are about to install on their PC has not been tampered with, and is also required by recent security additions in Windows XP Service Pack 2, as well as Windows Vista. Of course, the InstallAware 6 IDE itself can also code sign the setups it creates directly at build time.

Database Validator



The database validation tool scans MSI files produced by InstallAware and verifies they are fully compliant with the requirements of the Designed and Certified for Windows logo programs. These coveted logo programs have strict requirements which are enforced by Microsoft and their certification partners such as VeriTest. While InstallAware produces setups that are automatically logo compliant, and require no additional effort or awareness on part of the developer for adherence to logo guidelines, this tool is available for you to double-check that the produced MSI files will pass certification tests.

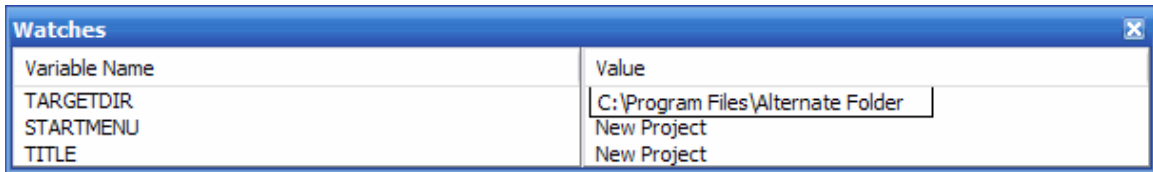
Desktop Layouts



The InstallAware 6 IDE offers the capability to save/load desktop layouts under different names, and also remembers the different layouts associated with each of the visual and MSIcode views. For instance, you could dock the variable watches window to the main IDE window, call it the debug layout, and use that layout whenever working in the MSIcode view.

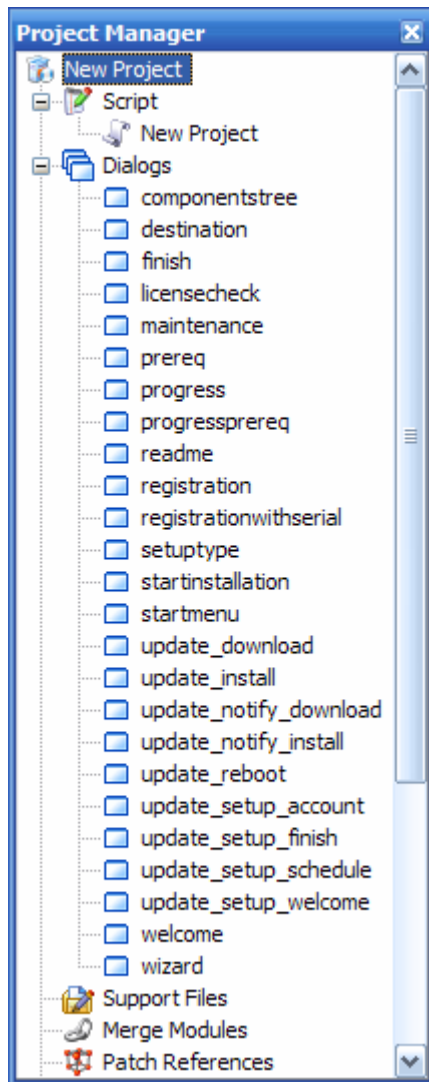
The main IDE toolbar also features a convenient drop-down that can be used to rapidly switch desktop layouts whenever working in the InstallAware IDE. The IDE associates different desktop layouts with the visual and MSIcode views available in InstallAware, hiding and displaying the various windows that are appropriate to the active context.

Watches Window



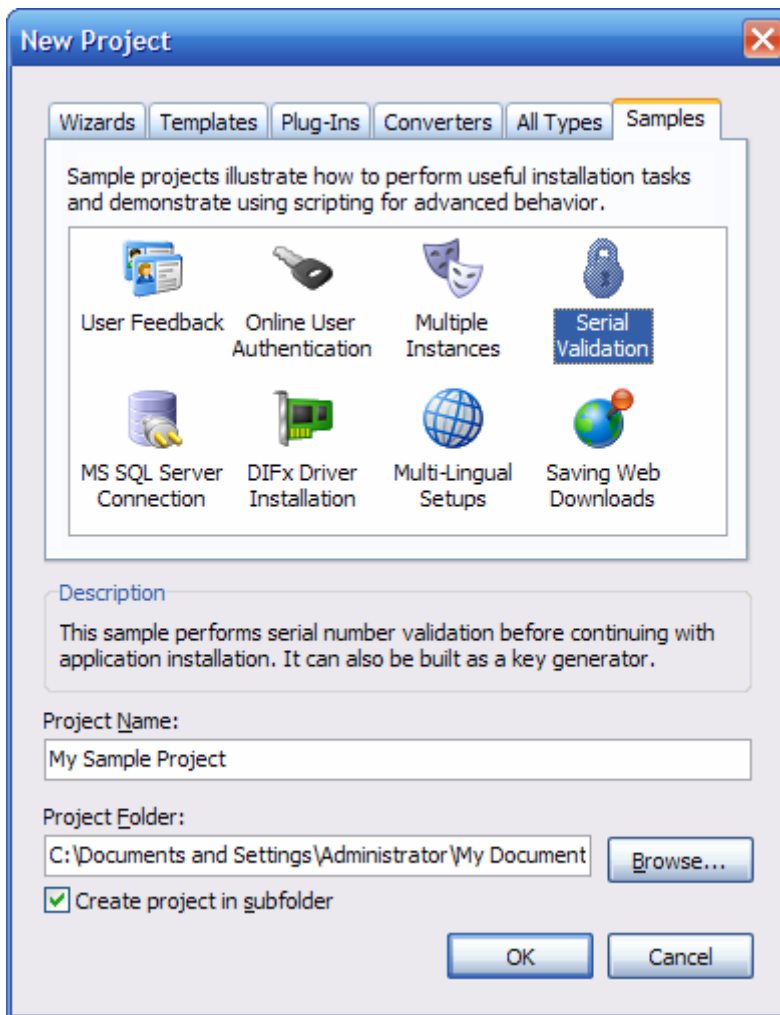
The Watches window lets you view the state of setup variables during installation, and also override the values of variables during debugging. Watches are a very helpful debugging aid.

Project Manager Window



The Project Manager window hierarchically displays the files that are part of your setup project, including your MSIcode script, include scripts, wizard dialogs, support files, merge modules, and references to other installers for patch creation. You may rapidly add/remove/modify any of the files that are a part of your project here. Docking the Project Manager to the main IDE window will also be helpful when working in the code view.

InstallAware 6 Wizards, Templates, Converters, Samples

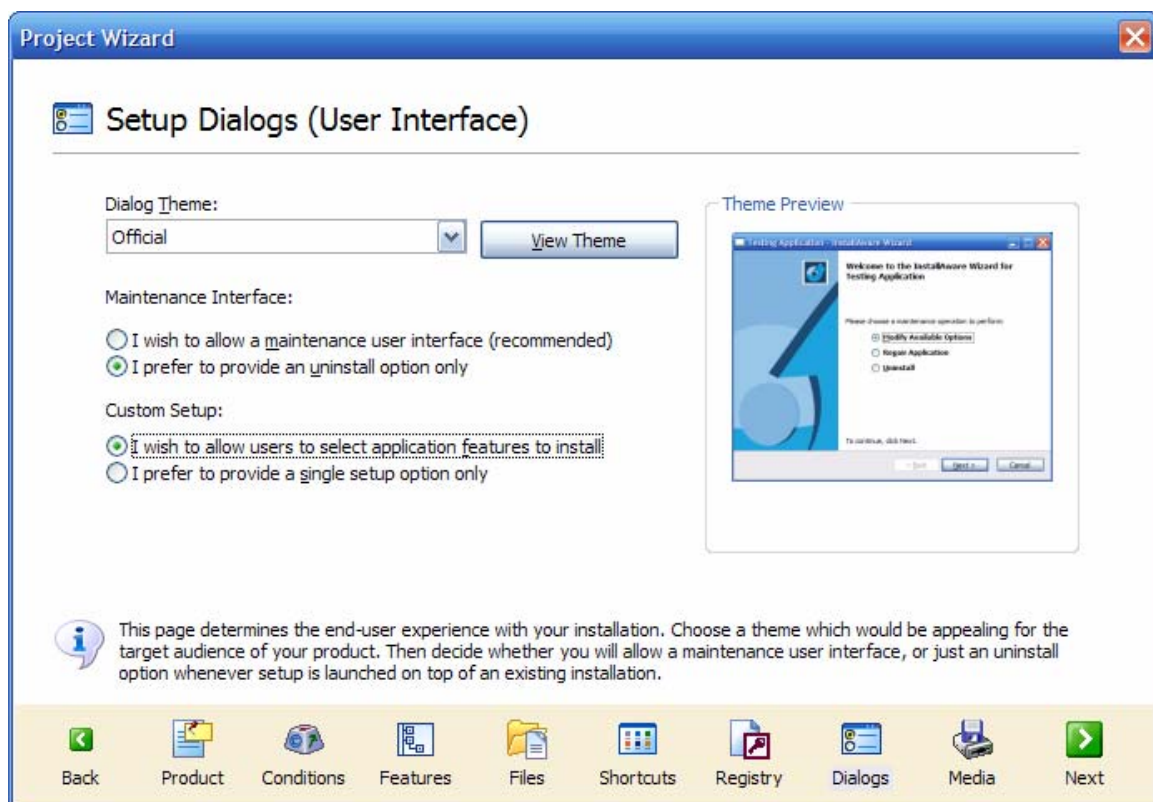


Wizards are small applets that help you to quickly create the projects, dialogs, and files that you use in InstallAware 6. Templates are pre-built setup projects ready to be customized further. Converters import setup projects built in other environments into InstallAware as ready-to-build projects. Samples illustrate performing useful tasks during an installation, such as capturing user feedback as an integral part of the installation. The InstallAware 6 IDE offers a variety of each to help you get started faster with creating your setups.

Default Project

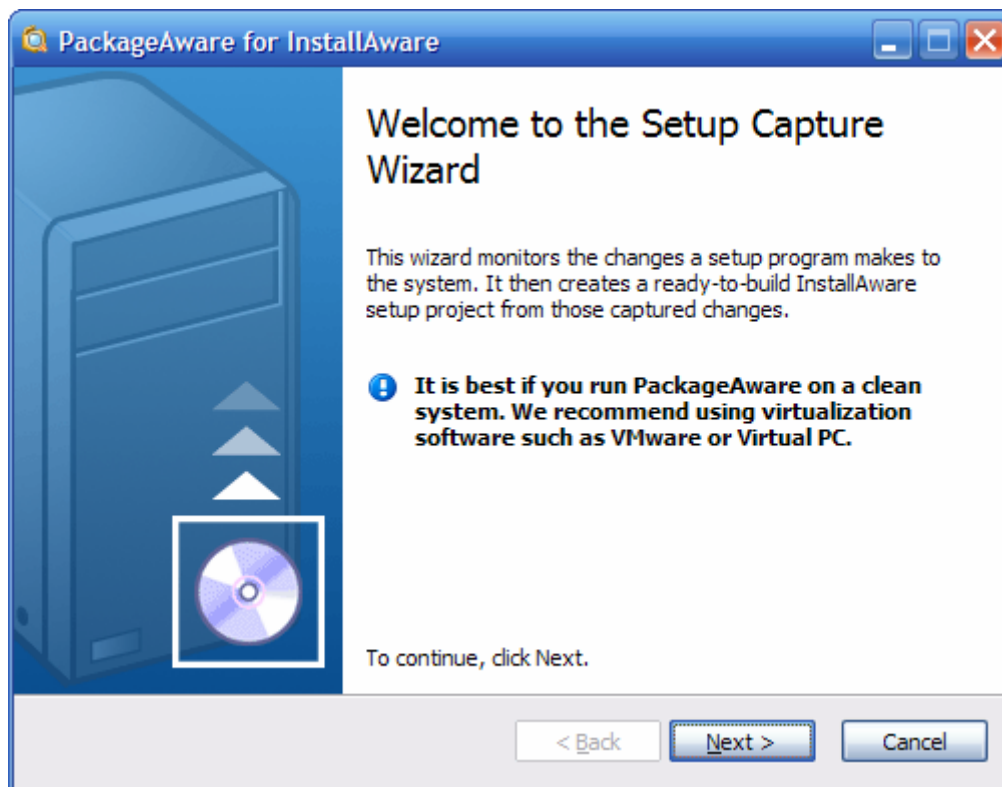
The Default Project creates a pre-built setup for a typical application. Just add in your files, registry keys, and build your setup. The MSIcode generated by the Default Project option contains all necessary scripting logic for proper setup behavior and extensibility, supporting Application Runtimes, Application Updates, System Pre-Requisite Checks, and many more.

Project Wizard



The Project Wizard guides you step-by-step, asking you specific questions related to your setup when creating a new installation project. Setups created by the Project Wizard have all the scripting logic in place for one-click patch creation, web media blocks, pre-installing various application runtimes, advertised installs, auto upgrades, and more advanced features.

PackageAware



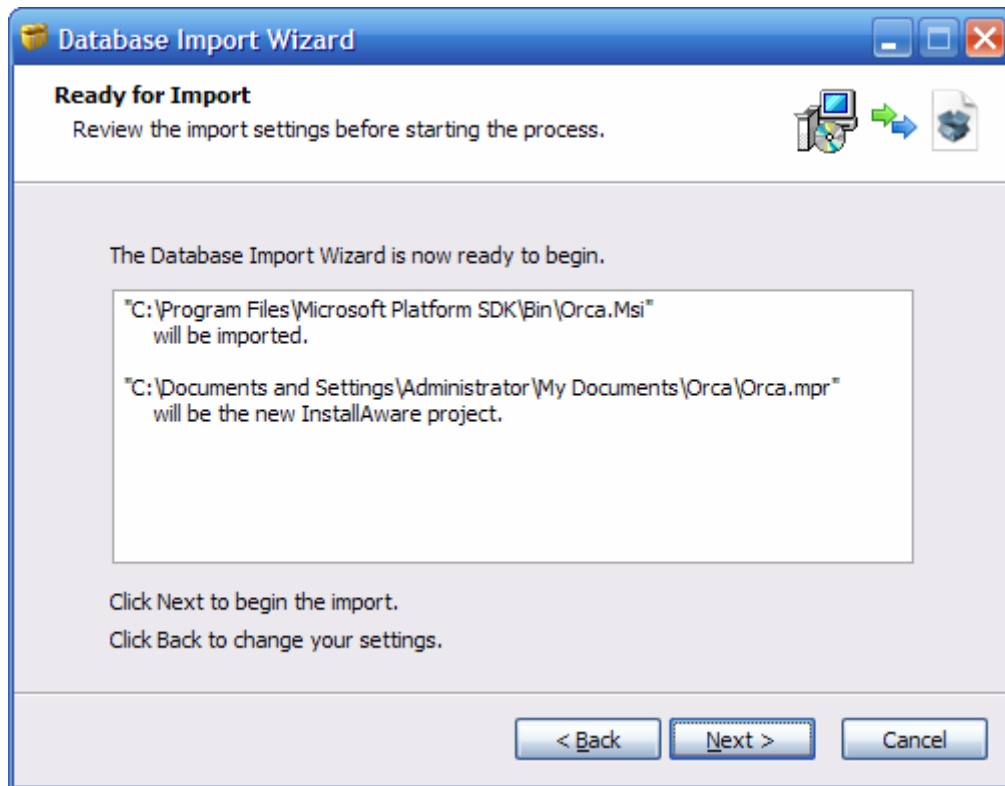
InstallAware 6 includes PackageAware. PackageAware may be used to convert legacy installations, built without InstallAware and/or Windows Installer technology, to a Windows Installer installation powered by InstallAware technologies. PackageAware works by scanning the system before and after an application is installed. It then compares those two scans and encapsulates the delta in a ready-to-use InstallAware installation project, complete with a script that you may further customize as necessary. PackageAware provides an effortless migration path to Windows Installer/InstallAware and also a clever way to monitor changes made to the system by installations, or other types of software applications.

New to PackageAware for InstallAware 6 are enhanced file and registry exclusion lists. These enhanced lists contain information on files and registry keys that are known to be safe to ignore during setup capture. This helps produce setup projects with virtually eliminated clutter, capturing only the real data that pertains to the installations you are converting – removing the “noise” generated by the operating system and other

background software processes running during your capture. With the new exclusion lists, your post-capture package tuning tasks are minimized, if not eliminated altogether.

Of course, setups based on Windows Installer may be directly imported into InstallAware using the Database Import Wizard, which directly reads the open MSI/MSM database format. Additionally, Windows Installer setups may be executed as part of a master InstallAware setup, completely maintaining the original setup logic as intended by the setup author, and avoiding the hassles of setup repackaging. InstallAware will even capture and display the progress messages coming from the original MSI file, if so desired.

Database Import Wizard



The Database Import Wizard in InstallAware 6 enables you to import external Windows Installer databases – MSI setups or MSM merge modules – directly into InstallAware as a new installation project. The projects produced by the wizard are ready to build and may also be customized further to serve your custom deployment needs.

Templates

Several templates provide baseline points for your setup projects:

- **Blank Setup:** Creates an empty project with an empty MSIcode script. Useful for highly custom scripting actions.
- **Win32 Setup:** Creates a pre-built setup for a typical Win32 application. Comes complete with support for pre-installing the Windows Installer runtime. Also supports one-click patching, advertised installs, and auto upgrades.
- **.NET Setup:** Creates a pre-built setup for a typical .NET application. In addition to the features in the Win32 template, offers support for pre-installing the .NET runtime.
- **Java Setup:** Creates a pre-built setup for a typical Java application. Offers support for pre-installing the Java Runtime. Note that Java setups may only install Java applications onto Windows platforms.
- **CD Autorun:** Creates an MSIcode script that does not perform an installation, but instead functions as a CD Autorun application, typically displayed whenever a CD is inserted into the drive. Autorun applications provide useful product information, launch setup programs, documents, and contain links to product websites.

Plug-Ins

Plug-Ins offer ways to seamlessly extend the InstallAware 6 IDE. Each plug-in has full access to the state of the installation, and can modify the values of variables used in the setup script. Plug-ins appear as an integral part of the MSIcode script and can render their script line dynamically based on the chosen command options. Several components of the InstallAware 6 setup development framework are implemented as plug-ins and you may also develop your own plug-ins.

Any language capable of developing Win32 DLLs may be used for plug-in development. If the generated DLLs have runtime requirements, be sure to include the necessary runtime files directly with the plug-in DLLs. Two template projects – Visual C++ Plug-In and Delphi Plug-In – are available. These templates already declare all required DLL exports and are ready to be populated with custom behavior.

Of course, any Win32 DLL may also be called from the InstallAware script - any arbitrary function prototype may be declared and invoked at run-time.

Converters

InstallAware 6 includes various converters that import setup projects created in various setup authoring environments directly to InstallAware:

- **Convert Wise Logic:** Converts a project built using Wise Installation System, or the custom scripting solution of Wise for Windows Installer. The entire setup script is migrated directly, since code forms are highly compatible. However because of slight differences in technology, the converted script will need to be touched up before it functions as a working installer. This option is most useful when the script to be converted contains highly customized installation logic which needs to be preserved. This option may also be used to convert blocks of code, which may then be copied and pasted into a larger InstallAware setup project.
- **Convert Wise Setup:** Converts a project built using Wise Installation System, or the custom scripting solution of Wise for Windows Installer. Unlike the “Convert Wise Logic” choice, this does not convert the script directly, but the installation itself; so that the resulting InstallAware project may be built and used immediately as a functional installation without needing to be touched up. This option may discard some of the advanced scripting that may be present in the original script.
- **Convert Inno Setup:** Converts a project built using Inno Setup. The resulting setup may be directly built as a functional installer. The conversion may discard some of the advanced scripting that may be present in the original project.
- **Convert NSIS Setup:** Converts a project built using Nullsoft Scriptable Install System. The resulting setup may be directly built as a functional installer. The conversion may discard some of the advanced scripting that may be present in the original project.

Of course, the Database Import Wizard can convert any Windows Installer database (MSI setup or MSM merge module); regardless of the environment it was created in. PackageAware may also be used to convert any setup by way of monitoring the changes it makes to the system, regardless of the technology or environment used in creating the setup.

Samples

InstallAware 6 offers new sample projects which illustrate how to use scripting and dialog editing to perform highly customized installation tasks, all as an integral part of the main installation:

- **User Feedback:** This sample illustrates how to capture user feedback from wizard dialogs during setup, and submit it seamlessly as part of the main installation. The sample performs online user registration during installation. It asks the user for feedback during an uninstallation which is also submitted online.
- **Online User Authentication:** This sample illustrates how to authenticate users on-line before continuing with the main installation. It also protects the setup with a global password, in addition to the on-line authentication process. When used with WebAware Installations, since the web media blocks that contain the critical product files won't even be downloaded until user authentication completes successfully, unauthorized users will be unable to obtain vital product files at all, even if they know the global setup password. This sample may also be easily extended to define application features on-the-fly, so users are able to see and install only the product features which they have been licensed to use. This makes the sample an ideal starting point for building a single installer that can install multiple product editions – such as Trial, Express, Professional, and Enterprise versions. By defining, displaying, and dynamically downloading only the live-authenticated features of your application at run-time, you may build a single, secure, unified installer with none of the overhead of maintaining different setup projects for the different editions of your product.
- **Multiple Instances:** This sample illustrates how to install multiple instances of the same version of an application. Typically, once an application has been installed, running the setup again allows only maintaining or uninstalling the existing copy. The first time this sample is run, it performs a normal installation. The next time the sample is run, it offers a choice to maintain/uninstall the existing instance, or to install a brand new copy of the application. All installed instances may later be maintained and removed from the Control Panel. New instances may only be installed from the main setup (and not from the Control Panel). While Windows Installer does not permit multiple instance setups on Windows 95/98/ME, and requires instance transforms on Windows NT platforms together with Windows Installer 3, InstallAware makes it very easy to create multiple instance installers that run on every Windows version, require only Windows Installer 2, and work without using instance transforms, as illustrated by the simple MSIcode scripting employed in this sample. Multiple instance setups are especially useful for server installations,

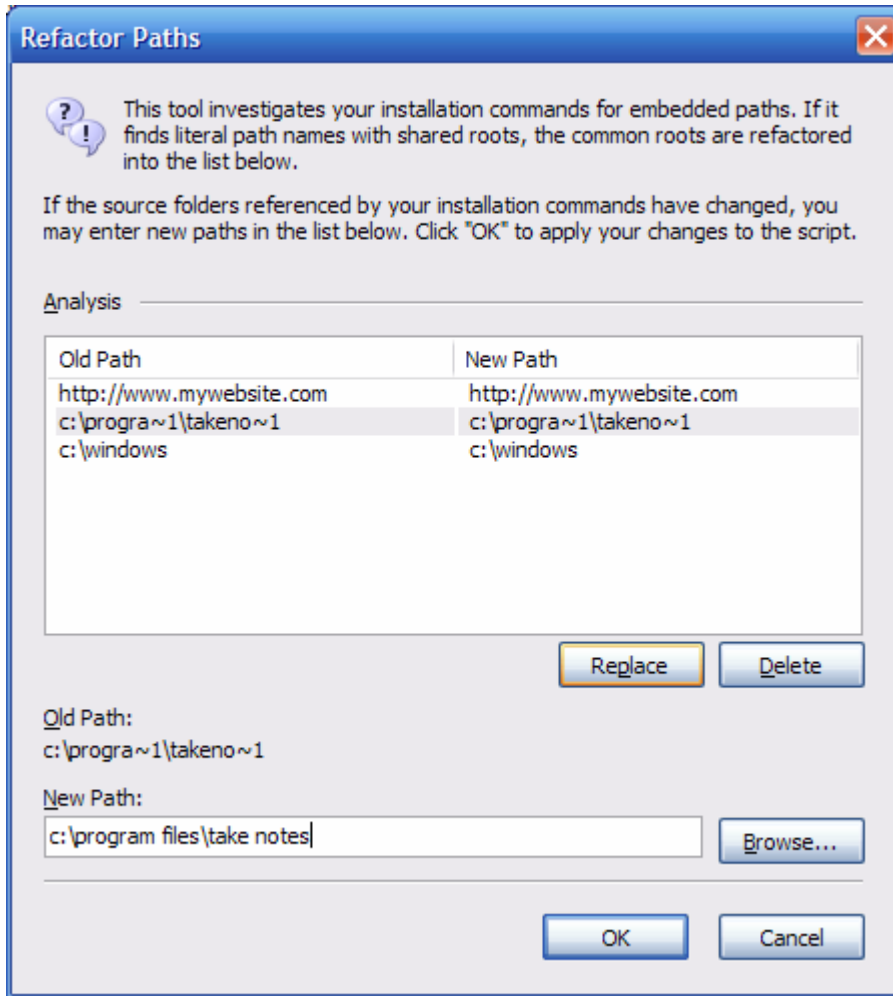
such as installing multiple copies of a SQL server application, with each instance offering access to different databases with different configuration, security, and user settings.

- **Serial Validation:** New in InstallAware 6, this sample demonstrates generating and validating a serial number in the MSIcode script before proceeding with the main installation. The provided Serial Number Generator script generates unique serials from the user name and company name entered during the installation – so you are not limited to a pre-defined range of serial numbers, and can produce unique serials for each end-user. The sample also illustrates conditional compilation using Compiler Variables, and can be built either as a setup, or as a key generator that generates serial numbers for user name and company name combinations.
- **MS SQL Server Connection:** New to InstallAware 6, this sample provides pre-built dialogs and logic that capture Microsoft SQL Server connection settings from the end-user, and then validate them in a connection attempt. The sample displays progress dialogs while validating server connections.
- **DIFx Driver Installation:** This new sample demonstrates the new driver installation capabilities in InstallAware 6. The sample first installs the files necessary for an imaginary “toaster” device, and then simulates attaching a virtual “toaster” to the system. Windows Plug-and-Play then automatically detects the attached “toaster”, and transparently installs the “toaster” device drivers.
- **Multi-Lingual Setups:** This sample illustrates how to switch setup languages at runtime using MSIcode scripting. It also displays a customized language selection dialog, which may be used in place of the standard language selection box displayed by the InstallAware setup loader.
- **Saving Web Downloads:** This sample illustrates how to save downloaded web media blocks and the main setup file in a folder of the user’s choosing. While downloaded web media blocks are automatically cached during a setup, and survive broken installations and reinstalls (so they do not need to be downloaded again), they are no longer available once the product has been uninstalled. By saving web media blocks and the main setup in a reinstallation folder, this sample shows how to build WebAware Installations that can be later used to run without any Internet connection, once the initial setup has run to completion. All the end-user has to do is backup the reinstallation folder and run setup from that folder at a later time.

Each sample contains plentiful comments, describing the MSIcode scripting commands and customized setup flow used in the sample. All samples may be used as starting points and customized freely to suit your particular installation requirements, without needing to reinvent the wheel and leveraging the already developed scripting logic and dialog customizations. Blocks of code may also be copied directly into your existing scripts, and your custom dialogs similarly updated, if you prefer to add the demonstrated features to setups that you have previously created.

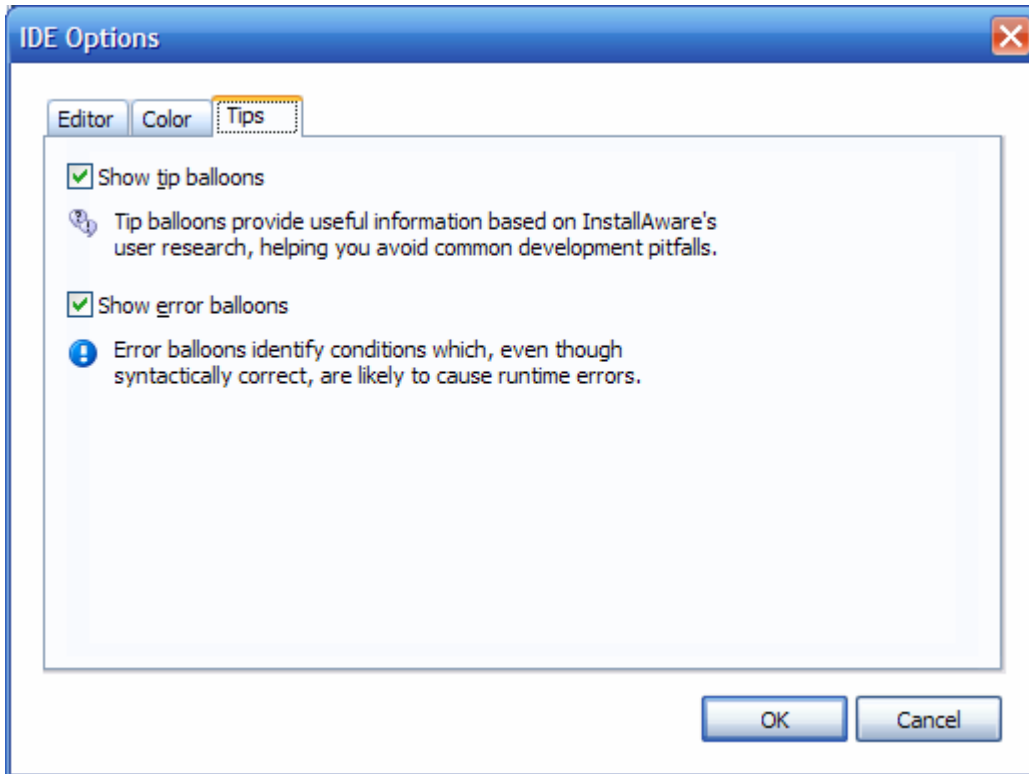
Refactoring

When more than one developer is working on a setup project, or setup projects are moved between machines, it often becomes necessary to change the source paths referenced by the project.



The Refactor Paths tool offers a very convenient mechanism to update embedded paths used in your setup project. The tool intelligently analyzes your project and extracts all hard-coded path names. It then obtains the common root paths and displays them for you to modify, all in a single pass. The Refactor Paths tool supports refactoring both URLs and file system paths.

IDE Options

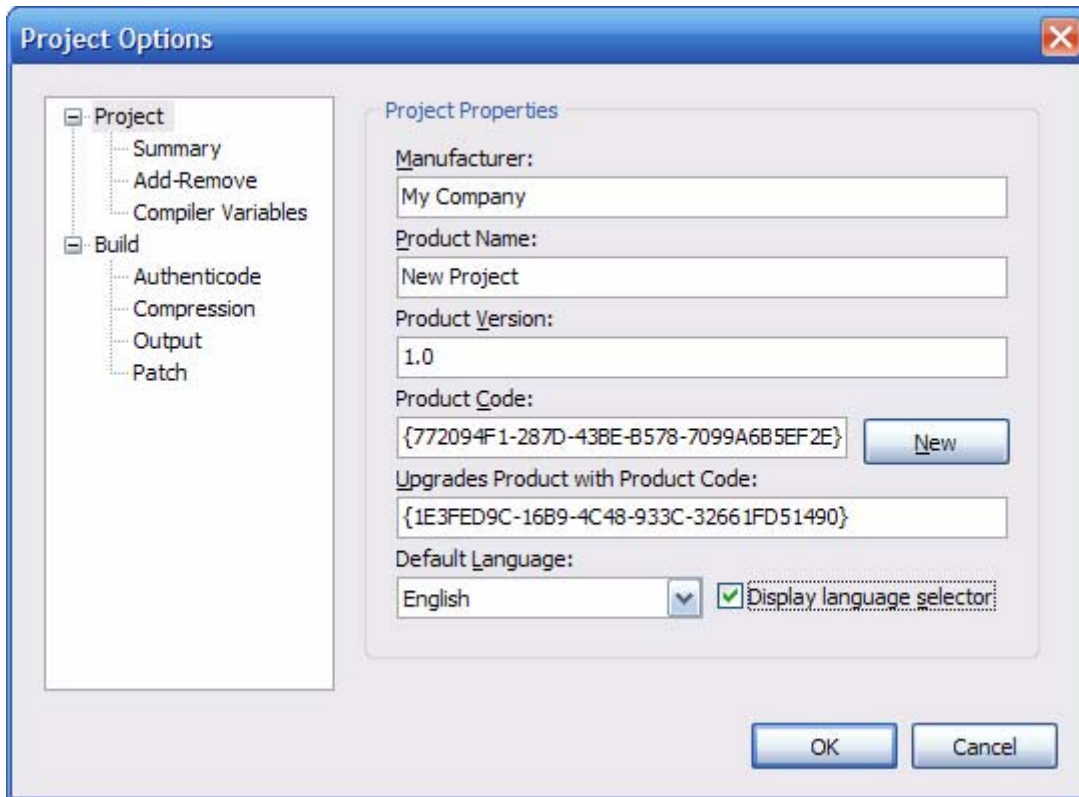


The IDE Options window lets you customize the font used for the MSIcode script editor, and enable or disable the code folding that is used when displaying your MSIcode script. You may also change the syntax highlighting used for the different types of scripting commands that are available in the InstallAware MSIcode view. For instance, by default, compiler directives are highlighted in gray, whereas scripting commands are displayed in black.

New in InstallAware 6, this window also lets you enable or disable the helpful tips that are shown while editing MSIcode commands and properties. Warnings highlight issues most likely to lead to compile time/runtime errors, and tips provide general helpful information.

Project Options

The Project Options window lets you customize your project settings and build options.



Project settings include the name of your project, how your application appears in the Control Panel Add-Remove programs applet, the summary database information that is recorded directly into the MSI installation database, the compiler variables together with their default values, the custom setup icon, and more.

Build options include the output folder and default file name for your setup, the package deployment mode (uncompressed, compressed, WebAware), authenticode code signing settings, compression levels used in compressing setup data, optional data protection using strong AES 256-bit encryption, and patch creation options.

Scripting Aware

InstallAware[©]
6



MSIcode

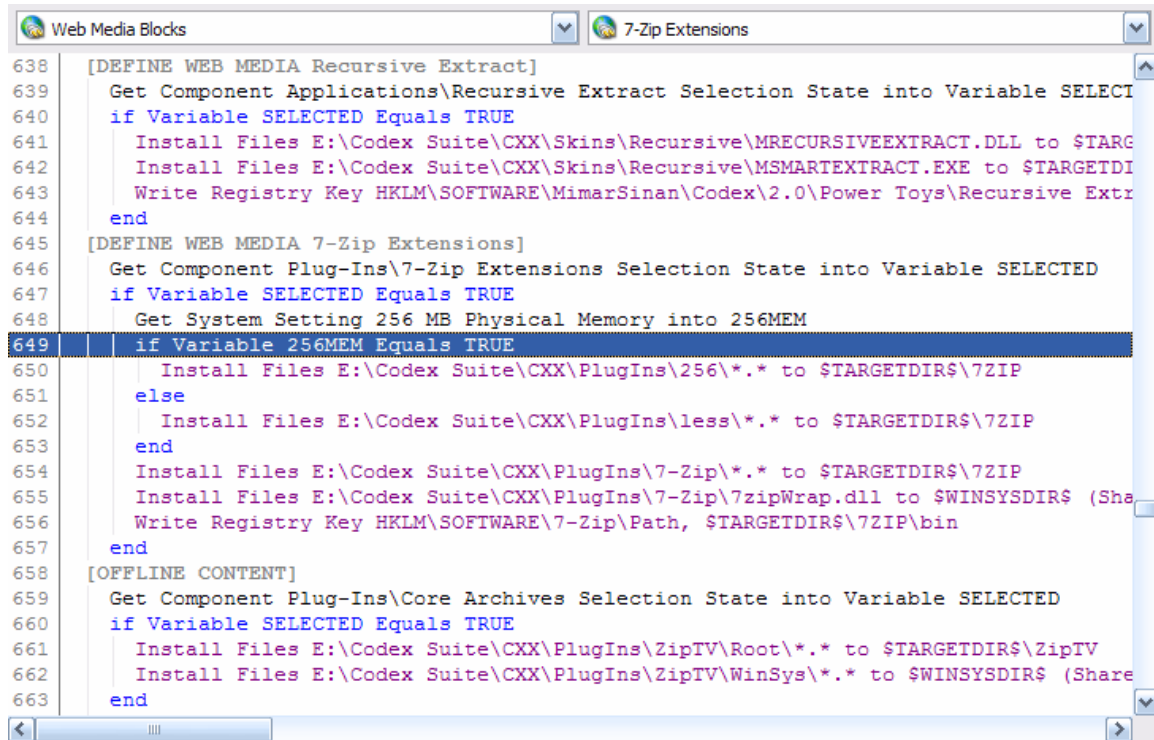
One of the key strengths of InstallAware 6 is the unique architecture it employs for creating Windows Installer packages.

While all other setup tools for Windows Installer provide high level visual editors for creating Windows Installer packages, none of them offer a native scripting capability for Windows Installer. Any scripting that is offered either requires an external scripting engine, which needs to be pre-installed and thus adds an extra layer of complexity to setups; or executes directly through another scripting technology which does not require to be pre-installed before use.

Even though this approach is suitable, it bifurcates installation projects into two disparate sets – one that goes through Windows Installer, and another that goes through the third party scripting engine (which may or may not need to be pre-installed). Separate editors and environments must be mastered for these two independent aspects of setup projects, and this increases the burden on the developer. It also complicates the development process as the two separate parts can interact with each other in unexpected ways and have unexpected dependencies. InstallAware has addressed this problem since its very first release in a conceptually clean, technically sound manner.

Types of MSIcode Commands

InstallAware 6 presents a single, unified MSIcode script which effectively deals with the bifurcation problem. The MSIcode script is the heart of the installation and has complete control over what happens – and when.



```
638 [DEFINE WEB MEDIA Recursive Extract]
639 Get Component Applications\Recursive Extract Selection State into Variable SELECTED
640 if Variable SELECTED Equals TRUE
641     Install Files E:\Codex Suite\CXX\Skins\Recursive\MRECURSIVEEXTRACT.DLL to $TARGETDIR
642     Install Files E:\Codex Suite\CXX\Skins\Recursive\MSMARTEXTRACT.EXE to $TARGETDIR
643     Write Registry Key HKLM\SOFTWARE\MimarSinan\Codex\2.0\Power Toys\Recursive Extract
644 end
645 [DEFINE WEB MEDIA 7-Zip Extensions]
646 Get Component Plug-Ins\7-Zip Extensions Selection State into Variable SELECTED
647 if Variable SELECTED Equals TRUE
648     Get System Setting 256 MB Physical Memory into 256MEM
649     if Variable 256MEM Equals TRUE
650         Install Files E:\Codex Suite\CXX\PlugIns\256\*. * to $TARGETDIR$7ZIP
651     else
652         Install Files E:\Codex Suite\CXX\PlugIns\less\*. * to $TARGETDIR$7ZIP
653     end
654     Install Files E:\Codex Suite\CXX\PlugIns\7-Zip\*. * to $TARGETDIR$7ZIP
655     Install Files E:\Codex Suite\CXX\PlugIns\7-Zip\7zipWrap.dll to $WINSYS$ (Share)
656     Write Registry Key HKLM\SOFTWARE\7-Zip\Path, $TARGETDIR$7ZIP\bin
657 end
658 [OFFLINE CONTENT]
659 Get Component Plug-Ins\Core Archives Selection State into Variable SELECTED
660 if Variable SELECTED Equals TRUE
661     Install Files E:\Codex Suite\CXX\PlugIns\ZipTV\Root\*. * to $TARGETDIR$ZipTV
662     Install Files E:\Codex Suite\CXX\PlugIns\ZipTV\WinSys\*. * to $WINSYS$ (Share)
663 end
```

The MSIcode script contains several different classes of commands:

- Comments: As found in all programming environments, comments are hints for the developer.
- Flow Control: Commands that direct program flow into alternate branches based on the evaluation of conditions.
- Plug-In: Custom plug-in provided commands.
- Directive: Directives to the MSIcode compiler, such as compiler variables, conditionally including/excluding certain types of code, and web media block declarations.
- Windows Installer: Commands that have direct correlates in the MSI database that is created at build time. Most commands that actually install the application (as opposed to, for instance, displaying dialogs and controlling program flow) fall in this category.
- Modify System: Commands that apply the pending changes to the system (an installation, advertising, patching, or an uninstallation).
- Label: Labels for use in directing program flow control.
- Statement: Any scripting command that either obtains system information, or makes direct, immediate changes to the system without going through the Windows Installer engine.

If you are a developer familiar with other Windows Installer authoring tools, you may find it easy to think of every command type, except Windows Installer commands, as custom actions. In effect, they are custom actions that are executed by the InstallAware engine. And, InstallAware also achieves conditional flow for your Windows Installer commands. The two combined together in a single setup script effectively eliminates the bifurcation problem with zero technical/conceptual overhead. Plus, the need to author and schedule custom actions – a very error prone, innately complex task – is eliminated, thanks to the richness of the MSIcode commands, and the wide range of pre-built installation behavior they cover.

Achieving Conditional Program Flow with Windows Installer

Windows Installer does not permit conditional program flow – that is, the developer may not create setups with execution logic like the following:

```
If Variable X is True then
  Install File Set A
Else
  Install File Set B
End
```

Windows Installer does permit attaching conditions to each component that installs, however this is a very inconvenient and counter-intuitive way of achieving conditional program flow, and in most cases is not a functional equivalent of conditional program flow either.

InstallAware, with its unique MSIcode technology, makes conditional program flow possible. Inside your MSIcode script, you are free to use as many Windows Installer commands as you like, and you may enclose those commands inside as many nested conditional commands as you want, directing program flow according to your installation requirements. This flexibility lets you focus on the actual logic of your setup, and gets your application installed correctly on a diverse array of hardware and software platforms, without having to worry about the implementation details on the Windows Installer side.

When you build your setup, InstallAware does several things behind the scenes:

- It parses your MSIcode for Windows Installer commands, and populates the necessary MSI table structures for them, and
- It attaches a unique condition to each Windows Installer command.

At runtime, as the InstallAware setup driver runs your MSIcode, when it comes across Windows Installer commands, it sets their unique condition to “execute”. And when the InstallAware setup driver encounters a Modify System command, it applies all pending installation (or uninstallation) changes to the system. This mechanism constitutes the heart of MSIcode, and very effectively creates the effect of conditional program flow on the Windows Installer platform, embracing and extending MSI technology to do what could not be done before. MSIcode will save you countless hours of development and testing time when building setups that require complex installation logic.

A Single Script for Installs, Maintenance, and Uninstalls

Another time saving feature offered by MSIcode technology is having a unified script for the entire install/maintain/uninstall cycle. If parts of an application are to be removed during a maintenance operation, and other parts added, absolutely no additional scripting code is required to install the new parts, or remove the old parts. And of course, no scripting code is required to remove all installed parts during an uninstallation.

A script with manual coding for the install/maintain/uninstall cycles could look like the following:

```
If Installing or Maintaining
  If Feature A is Selected
    Install File Set A
  If Feature B is Selected
    Install File Set B
End

If Maintaining
  If Feature A is Not Selected
    Remove File Set A
  If Feature B is Not Selected
    Remove File Set B
End

If Uninstalling
  Remove File Set A
  Remove File Set B
End
```

With InstallAware, you have a much simpler script that does everything in one place. If parts of your application are no longer needed, InstallAware automatically removes them (during either a maintenance operation, or an uninstallation) – no manual coding is necessary. If new parts of your application must be

installed, InstallAware automatically adds them. All powered by the same block of code – including the first time install:

```
If Uninstalling
  Modify System: Apply Uninstall

If Installing or Maintaining
  If Feature A is Selected
    Install File Set A
  If Feature B is Selected
    Install File Set B
  Modify System: Apply Install
End
```

When you compare the two code listings above, you immediately see that MSIcode, through its unified scripting model, saves you many lines of manual coding for the maintenance and uninstallation cycles of your application, while still operating on a highly customizable scripting model.

Limitations of MSIcode

Because each InstallAware setup is still based on Windows Installer technology, there are some unavoidable restrictions. Fortunately these are very few, and will not impact most setups. Moreover, these restrictions would apply to any setup based on Windows Installer technology – regardless of whether InstallAware or another Windows Installer tool was used.

The first limitation is that loops are not supported. While the InstallAware script contains constructs that allow you to code loops, if you execute a Windows Installer statement multiple times within a loop, only the last iteration of that command will actually run on the target system. Because InstallAware cannot determine at compile time the number of times a command located inside a loop may execute, it can populate only a single set of Windows Installer table structures for that command. At runtime, even though the command may execute multiple times, only its last iteration will have any effect. This is because only the last execution's setting of internal Windows Installer conditions will survive – the previous internal conditions will all have overridden one another.

The second limitation is that no changes are actually made to the target system when a Windows Installer command executes. You may think of this as InstallAware caching and queuing the command for deferred execution. The deferred commands are all executed at once when one of the Modify System commands are

called (Apply Install, Apply Advertised, Apply Patch, Apply Uninstall). Apply Install installs the application, Apply Advertised installs the application in advertised mode, Apply Patch patches a previously installed application, and Apply Uninstall removes an application. Because Windows Installer does not permit conditional program flow, the only alternative to this approach would be immediately applying changes to the system any time a Windows Installer command was called. While the InstallAware development team investigated this option early during development, it caused extremely severe drawbacks in execution time and was abandoned.

The impact of either limitation on installation development is very small.

For the first limitation, very rarely would commands that change the target system reside inside a loop. In fact, in the entire range of setup scripts developed by InstallAware Software Corporation, including internal development and external consulting projects, none of the setups had this requirement.

As for the second limitation, conditions that require setup commands to be applied immediately are also rare – they only occur under circumstances in which the setup flow branches based on the outcome of a preliminary setup command. If you do have a need for such branching, you will be relieved to learn that InstallAware allows you to call the Apply Install command more than once in your script, effectively sidestepping this limitation. The only drawback is that each time Apply Install is called, the Windows Installer engine is invoked – which performs a complete installation routine every time. Therefore calling Apply Install multiple times may degrade your installation speed, especially if the calls come after large blocks of file copy operations.

“Non-Windows Installer” MSIcode Commands

As extensive a setup engine Windows Installer is, it does not provide the total set of installation commands – and conveniences – that one expects to find in a complete, self-contained installation authoring environment. InstallAware 6 provides an extended, user customizable set of MSIcode commands which save you both time and effort in developing application installations.

Get System Information

Built-in MSIcode commands, which fall under the generic Statement category, offer ways to obtain extensive system information, including hardware properties (physical memory, display resolution), software properties (the running operating system, third party applications), user properties (privilege level, logged on user/domain/computer name), and more.

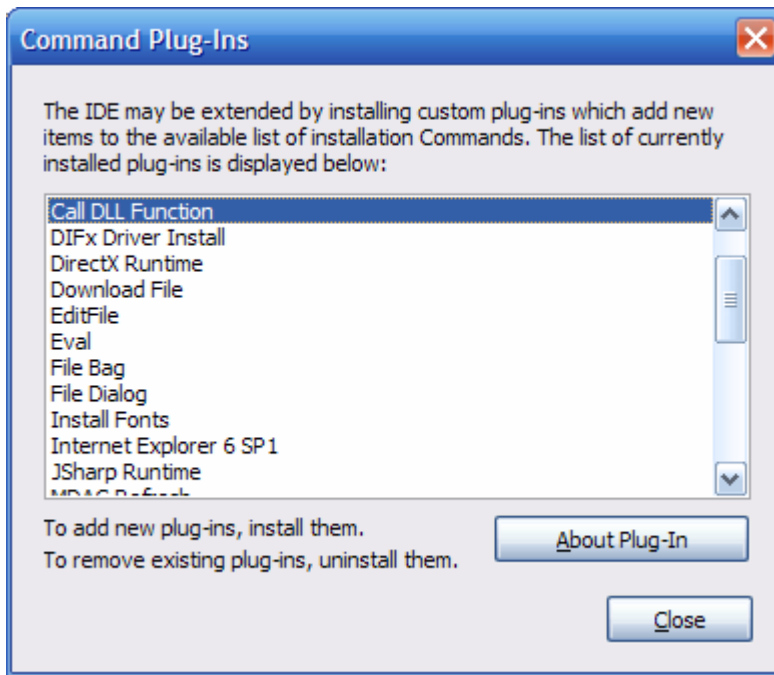
Advanced Install Actions

Statements are also available which perform advanced installation actions. For instance, you may create and update IIS websites and virtual folders. You can share folders or remove existing shares. You may set permissions on file system, registry, or other types of system objects. You may connect to a wide variety of database platforms and execute SQL scripts, including MS SQL, My SQL, and Oracle database servers (no client software is required on the target machine to connect to remote servers).

Run Programs, Call Dynamic Link Libraries

A very convenient way to extend the behavior of your MSIcode scripts is to call outside code – especially if you’ve already encapsulated such code into stand alone executable files or dynamic link libraries. InstallAware lets you run any external program, or call any arbitrary dynamic link library function with a variable list of parameters, all directly from your MSIcode script. The InstallAware 6 IDE also makes it even easier to run programs in your setup finish dialog, or before/after the main installation/uninstallation.

Plug-In Extensibility



Whenever you find that InstallAware MSIcode scripting falls short of meeting your needs (and such cases ought to be rare), you can always tap the power of your favorite programming language and directly code what you want to do in the development environment that you are most comfortable and productive in. InstallAware is plug-in extensible, and what's even better, each plug-in command also looks and works just like a native part of your MSIcode: It is visible in the script editor, can be copied/cut/pasted like any other MSIcode command, and has full access to the state of the installation, including reading from/writing to script variables.

InstallAware ships with several pre-built plug-ins that perform various tasks, and also includes two plug-in templates for plug-ins implemented using the Visual C++ and Delphi programming languages. Of course, you may develop your own plug-ins in any environment capable of creating standard Win32 DLLs.

Two-Way Integrated Visual View

One of the most time-saving features when scripting MSIcode, and one that has been greatly enhanced in the InstallAware 6 release, is the visual view that represents the installation actions taken by MSIcode graphically.

Just click the visual tab at the bottom of the InstallAware IDE window to switch to the visual view. The visual view intelligently parses your MSIcode script and also works with highly customized scripts. It contains 31 separate pages that represent different aspects of your installation, from files being installed to your setup dialog designs. Whenever you make changes in the visual view, it will automatically update the underlying MSIcode for you. It will do so without damaging your existing setup logic and correctly insert/remove MSIcode statements where applicable.

The visual view is a great time saver and helps you make the best of your MSIcode script – by letting you avoid custom scripting unless absolutely necessary.

Automation Interface

While scripting MSIcode in the InstallAware 6 IDE is a fun and enjoyable process, there may be times in which you need to programmatically emit MSIcode scripts without depending on the IDE. For instance, you may be building a large set of installers with very similar features but different branding/names for your network of resellers. Especially if you have a large number of resellers and installers to customize, this could very quickly escalate into a development nightmare.

With the automation interface in InstallAware 6, you get full programmatic access to MSIcode scripts. Directly from your own external programs, you may:

- Emit a complete MSIcode script, line by line
- Emit a setup project, with its own settings, dialogs, support files, and such
- Build a release, or a patch

This provides unprecedented flexibility in developing installers. Continuing our reseller scenario, you could build a website where each reseller could log on to their account, and upload their custom graphics and text for their brand/version of the product. Then they would click a “Build Installer” button which would immediately invoke the InstallAware automation interface, emit and build their setup, and deliver the download link. Any time you updated your software to a newer version, you wouldn’t have to worry about manually updating the many installers for your resellers either! You’d just let them know that you have a newer version available, and they could all log on to their accounts and generate their personalized installers – all delivered by the InstallAware automation interface.

This is just one example for a case where InstallAware automation takes a major burden off of your shoulders. There are countless other uses for this technology. Some InstallAware licenses even permit you to redistribute the automation libraries, so you can emit and build installers directly on end-user systems! For instance, consider a screen saver creator application – each screen saver it generates would also need an installer. Developing a home made installer for the screen savers produced would be a major undertaking. With a redistributable automation interface, you can simply call the InstallAware library functions and let InstallAware do what its best at – building setups – while you get to focus on your core expertise.

The automation interface is cleanly encapsulated inside Win32 DLLs which can be called from any Windows programming language. The automation interface also provides an ASP object which can be called from an Active Server Pages website.

The InstallAware Customer Site, which vends licenses to developers who have purchased InstallAware, uses the automation interface to create watermarked installers that contain the name of the individual or organization that purchased the license.

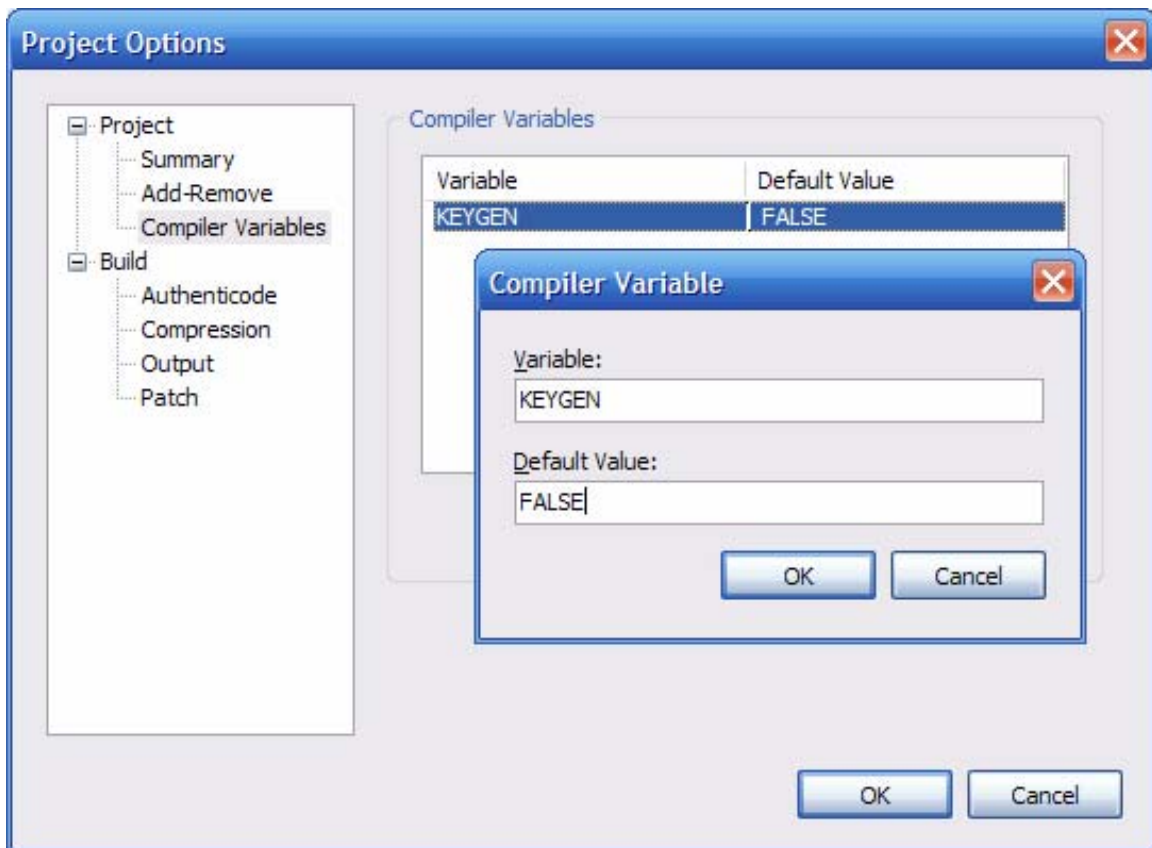
Compiler Variables

There may also be cases in which you need to customize your installers – but not as thoroughly as needing to use the automation interface. For instance, you may have several editions for a product, each with progressively more features and higher prices. When building the installers for these various editions, instead of coding multiple scripts for each, you would rather code and maintain a single script. In this case, using the automation interface would be overkill – another way to quickly customize the script becomes necessary.

Compiler variables are resolved at build time, and let you achieve the following:

- Include or exclude various parts of MSIcode based on compiler variable values
- Substitute compiler variables used in MSIcode commands with their literal values

Continuing our example with the various editions, you could use Compiler Variable If statements to conditionally include/exclude parts of your MSIcode, and hence, application files, based on the product edition. You could also use compiler variables directly in MSIcode commands for build time substitution of their values into the script (such as, the edition name).

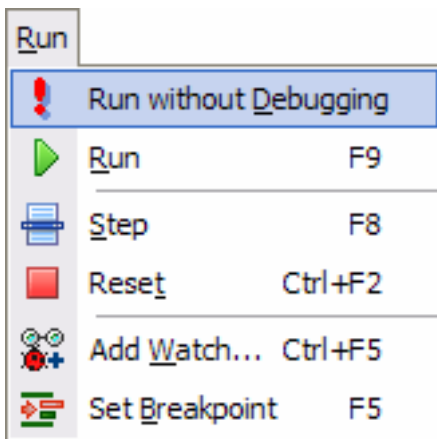


Compiler variables, and their values, are defined in the Project Options window. The values specified in the Project Options window may also be overridden while building from the command line, or via the automation interface. Coupled with the redistributable command line build interface that is provided with higher-end InstallAware editions, compiler variables may also be used to create customized installation

packages directly on end-user systems – again without having to use the automation interface to programmatically emit full setup projects from scratch.

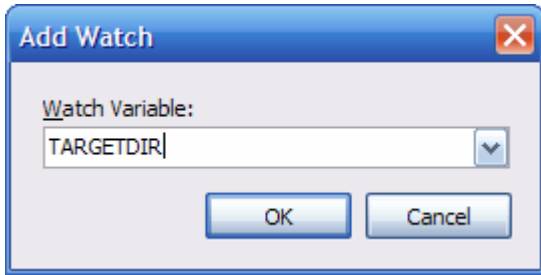
Integrated Debugging

Despite the beauty and power of MSIcode, or perhaps because of it, the need to debug the scripts you have developed in InstallAware will arise sooner or later. Along with the canonical MessageBox command that can be used to output values of variables to the screen, InstallAware 6 provides a visual, integrated debugger right within the main IDE.



The run menu contains all the debugging commands you need:

- Start debugging
- Step through MSIcode line by line
- End a debug session
- Add variable watches
- Set breakpoints on MSIcode statements



You may also override the values of variables while in a debugging session to test alternate routes of program flow.

Web Aware

InstallAware[©] 6



WebAware Installations (Partial Web Deployment)

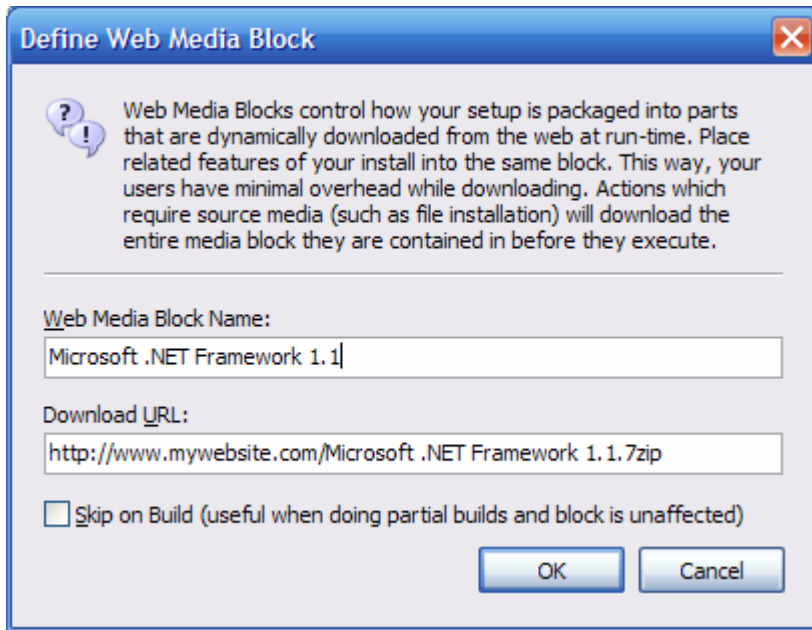
Another unique feature of InstallAware 6 is the possibility of partial web deployment.

With all other tools, setups that are web deployed reside entirely online. The end-user downloads a very small stub that contains only the setup driver. The stub then connects to the Internet and downloads the entire remainder of the setup from the Internet. This approach definitely saves time in some cases – for instance, the stubs don't download runtime components, such as the .NET runtime, if they are already installed on the end-user system. However, having to download the main application from the web as well, in addition to the runtimes, is an inconvenience at best. Even if the end-user system already has the required runtimes, the setup will still require and force an Internet connection.

The other alternative is to pack every runtime, along with the application itself, into a self extracting installer that contains everything. However, the downside to this approach is that while the setup is completely self-contained, and thus installs without requiring an Internet connection, it is bloated and may already contain runtimes that the end-user already has. Especially in the case of larger runtimes, such as the .NET runtime, this bloat can far exceed the size of the main application itself, and frustrate users in that they have to download and wait for what they already have on their system. InstallAware provides the perfect solution to this dilemma since its very first version: web media blocks technology.

Web Media Blocks

In InstallAware 6, setups are comprised of one or more web media blocks. A web media block is actually an MSIcode directive, very much like the compiler directives found in most programming languages. It tells the MSIcode compiler how the setup is to be split into multiple online and offline pieces.



A web media block directive has two fields: the name of the web media block, and its download URL. The name of the web media block determines the name of the output file that is created for that web media block. This file name will typically be of the form <block name>.7zip. The download URL indicates where the installer can expect to find the web media block file at runtime, and can point to any HTTP, FTP, or network location. The default file name of the web media block can also be modified in the download URL. As long as the actual web media block file exists at the given location, the download will succeed.

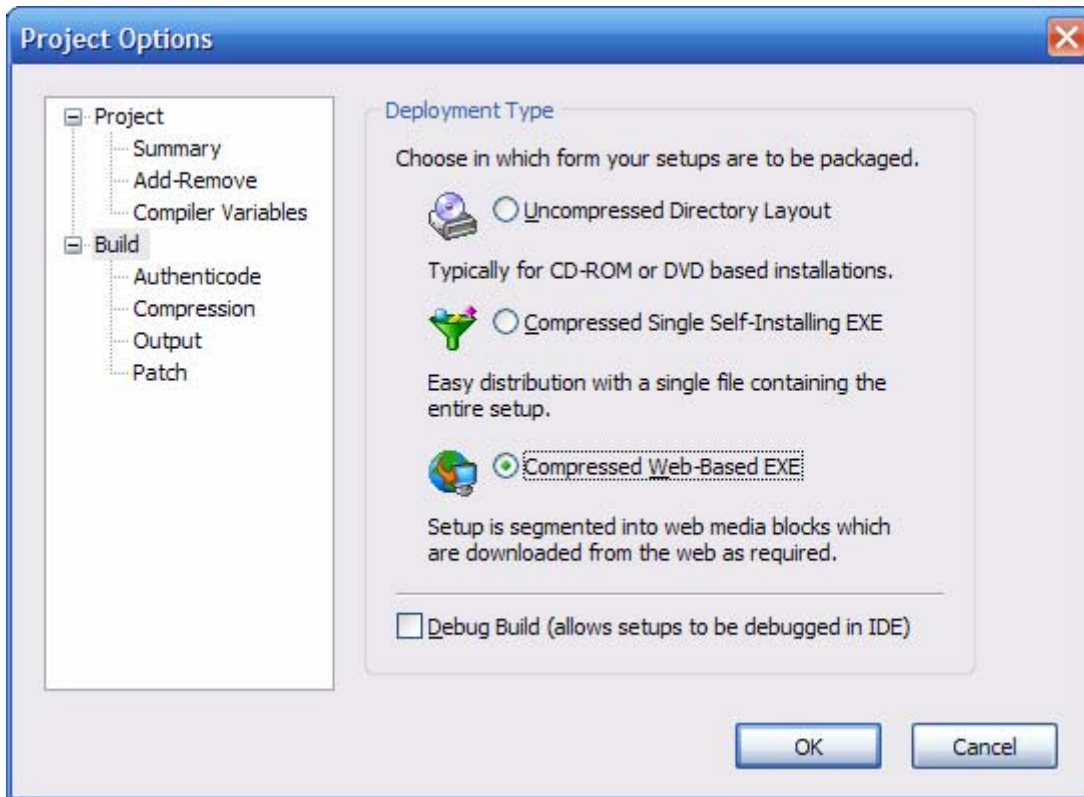
There is also a special kind of web media block, called the offline block. If a web media block statement contains an empty name, that web media block is considered to be an offline block, and the download URL field is disabled. The installer does not create a separate file for an offline block, and instead packages everything inside the offline block into the main setup executable (hence, the “offline” block).

When InstallAware 6 builds your setup, it parses your MSIcode and identifies which files belong inside which web media blocks. A web media block directive has effect from the point of its declaration in the script up to the next web media block directive. Each MSIcode command that requires source media, such as Install Files, becomes associated with the last declared web media block directive and the files for those commands are packaged into the web media block file defined by that directive, to be downloaded dynamically at runtime. You may declare an unlimited number of web media blocks in your MSIcode, and also use identical declarations more than once, in this way precisely describing the partial web deployment structure of your setup.

At runtime, whenever an MSIcode statement that falls inside a web media block executes, the installation engine attempts to download that web media block (if it has not been previously downloaded). If none of the statements that require access to source media (such as Install Files) inside a particular web media block execute, that web media block will not be downloaded. The download process is resilient and has the capability to resume after an interruption. Moreover, if operating within a corporate environment where proxy servers are used, the download process automatically detects correct proxy settings and works seamlessly.

WebAware Installation technology, only available with InstallAware, provides flexible, partial web deployment capabilities for the first time in setup development history. You may place each separate application runtime inside its own web media block, therefore preventing bloating the size of your main setup executable. You may also use this facility to separate rarely used or space consuming features of your application from the main setup file. And any part of your setup which is inside an offline block goes directly inside the main setup file, helping you build a self contained installer that works without demanding an Internet connection.

Splitting your application into web media blocks will not only save your users undue frustration, but it will also result in significant bandwidth savings for your business. Because large runtimes and optional features will be downloaded only when absolutely required, your servers will conserve bandwidth up to levels not possible before.

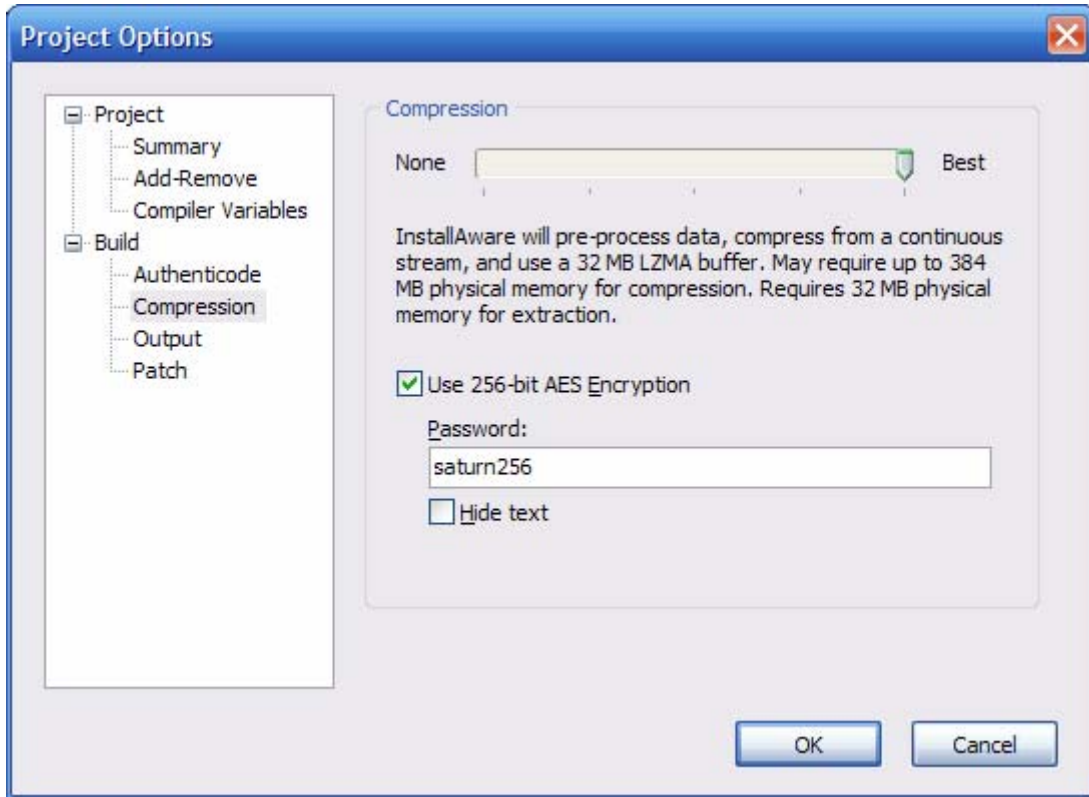


While every InstallAware MSIcode script will contain one or more web media blocks, the build process will generate web media blocks only if you are targeting the web (as configured in the Project Options dialog). If you are doing an uncompressed or compressed build, web media blocks will be excluded from the installation, and every part of your setup will be available as either a single monolithic compressed file, or a collection of uncompressed files and folders suitable for CD/DVD deployment. This adds an additional layer of convenience when you are targeting multiple types of distribution media and avoids the tedium of having to manually modify your setup project for each different output type.

Superior Compression

In tandem with its unique web media blocks technology, InstallAware 6 provides a revolutionary form of compression that works nothing short of miracles in reducing file sizes. InstallAware 6 expands on the abilities of its previous versions and goes even further in reducing download times and sizes.

New in InstallAware 6 is the ability to split compression tasks across multiple CPUs, CPU cores, or logical (hyper-threading) CPU units. Compression is computationally intensive, and parallel execution improves compression speeds by as much as 50% – saving you time, as well as delivering better compression ratios.



Compression settings are configured in the Project Options dialog, and five levels of compression are available, with a sliding scale from none to best. The best compression mode utilizes a very advanced data processing algorithm that pre-processes all files using Binary Call Jump Converters (version 2), thus increasing their compressibility; and then passes the optimized data streams through a LZMA encoder with a 32 MB compression buffer.

You may also protect compressed setups with a global password. Data is encrypted using strong 256-bit AES encryption for bullet-proof security.

InstallAware’s compression speaks for itself. The following table lists size savings for several popular application runtimes after having been processed by InstallAware:

Runtime	Compressed Size	InstallAware Size	Compression Savings
Windows Installer 2.0	3,446 KB	1,770 KB	49%
Internet Explorer 6	79,849 KB	47,110 KB	41%
Microsoft .NET 1.1	23,697 KB	12,338 KB	48%
Microsoft .NET 1.1 SP1	34,150 KB	11,403 KB	67%
Microsoft JSharp 1.1	6,764 KB	2,596 KB	61%
Microsoft .NET 2.0	22,959 KB	17,272 KB	25%
Microsoft SQL Express 2005	54,791 KB	33,036 KB	40%

Notice that the savings listed in the table above are over the already compressed sizes of the runtimes. In most cases, InstallAware reduced more than half in size what was already the compressed size of the runtime.

Similar compression benefits carry over to your own program files as well, helping you create the smallest possible installers available today.

Repacking Databases

While the examples above imply that it is possible to re-compress pre-compressed data, this is not the case. No matter how good a data compression algorithm is, because of the law of entropy in data compression theory, it cannot recompress data that has been compressed before, even if the prior algorithm was an inferior one. Therefore, InstallAware actually comes with de-compressed copies of the runtimes in the list above (among others), and it is these runtimes which it compresses to half the size of the original compressed sizes. But what about your own runtimes and program files?

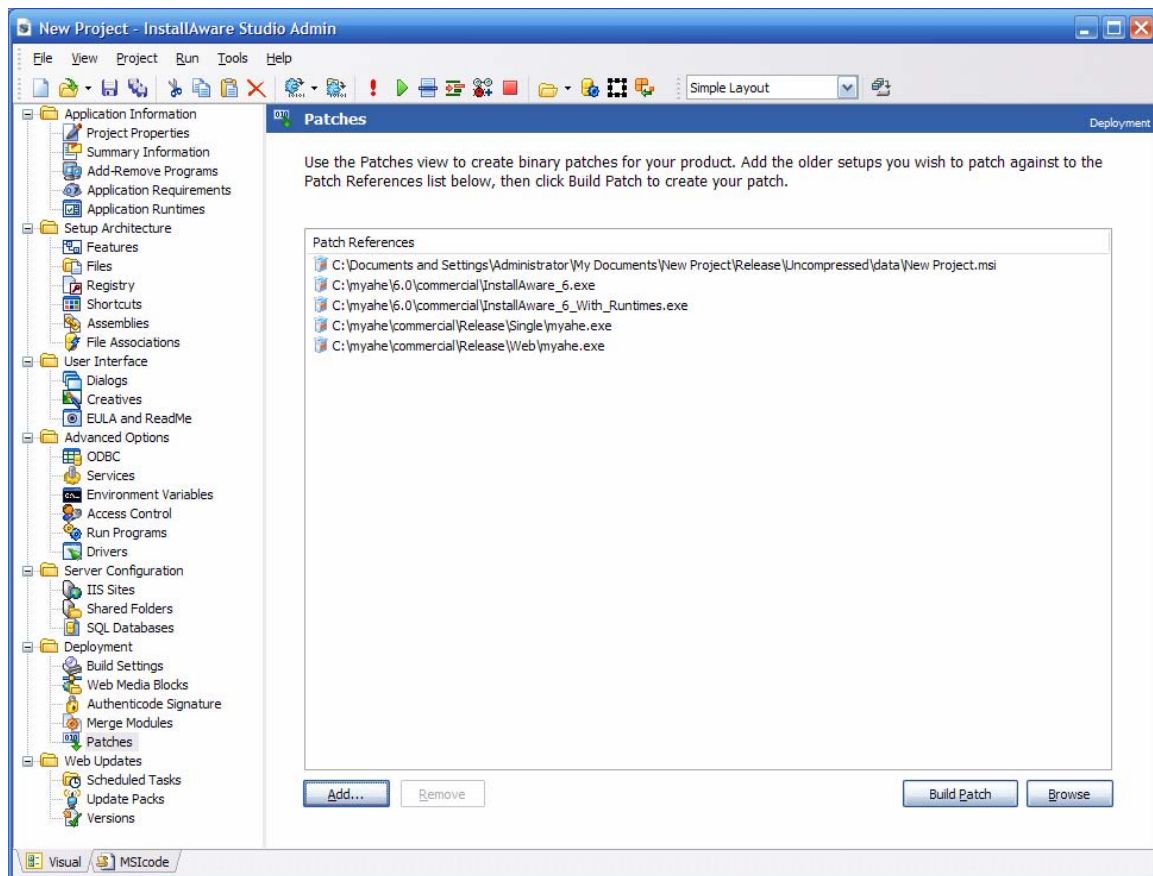
When you are trying to take advantage of the superior compression that is provided by InstallAware, you should make sure that none of the source files going into your setup are pre-compressed. For instance, do not use EXE-packers, which reduce your executable program size, with InstallAware setups – because while the EXE sizes will indeed be reduced with the EXE-packer, they will be reduced to even less of a size if you let InstallAware compress them from scratch.

There might also be cases in which you include MSI and MSM Windows Installer databases in your setup, to install additional components or dependencies. Unfortunately, MSI and MSM databases generally contain compressed data streams within them, and of course, even though the native Windows Installer compression is inferior (CAB-MSZIP/CAB-LZX), InstallAware will be unable to further compress this data because it has already been compressed.

InstallAware 6 provides a fully automated tool, the Setup Decompressor, which you can use to repack these installation databases. This tool works with MSI and MSM databases, as well as most Microsoft EXE setups. It updates the compressed data they contain with the exact same data in uncompressed form. This allows InstallAware's superior LZMA/BCJ2 algorithm to work miracles in re-compressing all the data that is found inside the Windows Installer databases. Whenever you add an MSI or MSM database or a Microsoft EXE setup to your installation, be sure to process it first with the Setup Decompressor tool.

One-Click Patching

No matter how good the compression you use, and how many web media blocks you split your setup into, the time inevitably comes when you need to issue an update to your end-users, and you just cannot afford to resend a complete installer. This is especially the case when a relatively minor number of files have been updated for a relatively small number of changes and bug fixes in your application, and you just want to update those files – instead of re-distributing the entire setup from scratch.

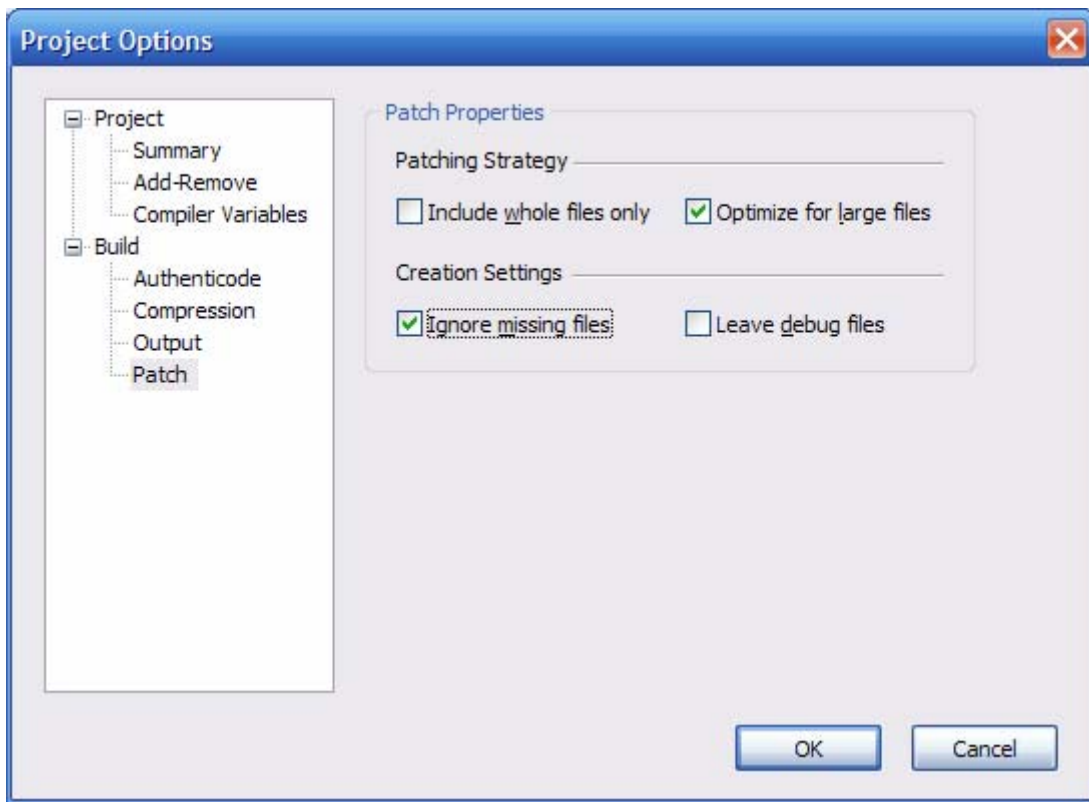


InstallAware 6 provides the unique one-click patching feature to address this need. To use one-click patching, first prepare your setup normally. Then, add the installers for the previous versions of your product to your setup project as Patch References (only add the installs that you wish to support patching against). When you click the Build Patch button, InstallAware does the following, all automatically, with no manual intervention or supervision on your part:

1. Builds the latest version of your application setup,
2. Extracts the referenced previous versions of your application setups (if compressed), to a temporary working folder,
3. Compares the current version of your setup with the previous versions,
4. Invokes the Windows Installer engine to generate a native Windows Installer patch file (.MSP), and
5. Repacks the native Windows Installer patch file using InstallAware's superior compression.

InstallAware takes care of all the dirty work in patch creation for you. Unlike other leading commercial products, no upgrade syncing, no patch compatibility testing, and no pre-flight deployment testing is required. InstallAware patches take truly one-click to create, and they just work, without complications.

The created patches contain the absolute minimum data and files that are required to upgrade your old versions to the newest version. This is made possible by binary/differential scans of the files in your different versions. Only the changes between the different versions are included in the patch. Even when patching very large applications, the output patch files will be very small – dramatically simplifying patch distribution and reducing your costs.



Further patch creation options, such as using full files instead of storing binary/differential patch data, are also available in the Project Options dialog.

Windows Installer Aware

InstallAware[©]
6



Windows Installer Functionality

InstallAware is based on Windows Installer, the software installation technology that is developed by Microsoft and is gaining ever-increasing acceptance. Windows Installer has always been strong in the corporate market because of its innovations in deployment and management, and it is also gaining increasing acceptance in the desktop market. The future is clearly heading in the direction of Windows Installer – and InstallAware takes you there with a minimum of effort today. Most large organizations no longer even consider applications that do not ship with a Windows Installer setup – the management overhead of deploying non Windows Installer setups onto thousands of corporate workstations is prohibitive.

InstallAware 6 adds even more support for Windows Installer features, and does so in the traditional InstallAware way – with no effort or expertise necessary on your part. While Windows Installer is very complex to learn and use, InstallAware provides crisp abstractions in its IDE and setup authoring process, isolating you from all intricacies of the underlying relational database and implementation details. InstallAware 6 extends these abstractions to the entirety of the Windows Installer domain – so today, with InstallAware 6, you leverage all the power of Windows Installer, with none of the cost.

Logo Certifiable Setups



One of the application requirements for earning the coveted Designed for Windows logo, as determined by Microsoft, is having a setup routine developed in Windows Installer. This requirement has two main parts:

1. The setup database (MSI file) must pass ICE (internal consistency evaluators) tests, and

2. The setup database must create advertised entry points for the application, in order to enable install-on-demand.

InstallAware MSI databases have always passed ICE tests. As provided by MSIcode technology, whenever an InstallAware setup script is compiled into an MSI database, it is automatically ICE compliant.

You may manually test the ICE compliancy of your setup databases at any time using the MSI Database Validation Tool. The tool contains specialized validation suites for Windows 2000 and Windows XP logo programs. It also runs on all Windows platforms (unlike most validation tools which fail to validate running under the Windows Server 2003 operating system).

InstallAware 6 also supports the second part of the logo requirement: advertising. InstallAware 6 provides several MSIcode commands which support the creation of advertised entry points in applications. The setup engine performs advertised installs of a product when these commands are used.

Advertising

Advertising is a concept in application installations that is unique to Windows Installer. Currently only Windows Installer based setups, including those created by InstallAware, support advertising. In a nutshell, the notion of advertising represents two capabilities:

- An installation mode, in which an application is not actually installed, but only appears to be installed; to be fully installed upon first use (install-on-demand)
- A way to verify the integrity of an application installation every time it is launched (self-healing installations)

Install-on-Demand

When we install an application, our indicators that the application is installed are shortcuts in the Start Menu, new file types which are now visible in Explorer with their custom icons, and the like. These are termed “application entry points,” and any time we actually invoke the application through one of these

entry points, the actual application executable is run. And of course, the application executable is already present on the system.

When an application is installed in advertised mode with Windows Installer, that is, the application is “advertised,” only the entry points of the application are created. Thus, to the ordinary eye, the application appears to be fully installed – the Start Menu items are all present, new Explorer file types are visible with their custom icons, and so on. However, during an advertised install, no files are actually copied. Advertised installs therefore dramatically accelerate installation time, and provide the appearance that the advertised product is actually installed, when in fact only the application entry points are installed – hence, the application is being ‘advertised’. When an advertised entry point is actually invoked, such as a Start Menu shortcut being launched or a file associated with the application being opened, the Windows Installer service automatically intervenes and immediately installs the required files, and then seamlessly launches the application.

This is called install-on-demand and is made possible with Windows Installer. InstallAware 6 fully supports installation-on-demand, and even better, does not require any changes to your script sources for advertising. When you use the Apply Advertised MSIcode command, InstallAware advertises the product. Compare this behavior to the Apply Install command, which would fully install the product.

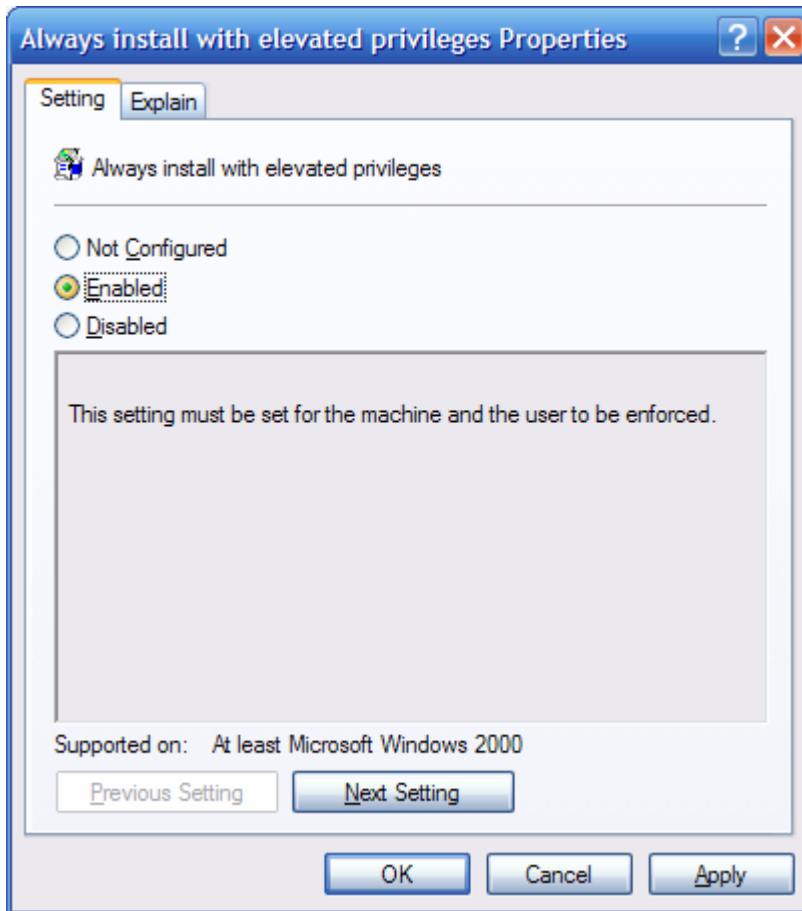
Self-Healing Installations

Another benefit of the advertising technology offered by Windows Installer is the ability to create self-healing installations. Windows Installer not only supports install-on-demand from application entry points, it also uses the application entry points as an opportunity to monitor the health of the application installation. In particular, if the target of a Start Menu shortcut or a file type is missing, Windows Installer will again automatically invoke and repair the installation, replacing the missing files.

This self-healing behavior helps assure the integrity of applications and provides an additional level of convenience for end-users who may have lost application files due to their own error, a system fault, or any other cause. When you use the MSIcode commands in InstallAware 6 that create advertised entry points, your installs automatically become self-healing.

Elevated Privileges

Operating systems based on the Windows NT kernel (Windows NT 4, Windows 2000, Windows XP, Windows 2003 and Windows Longhorn) allow setting permissions on file system and registry objects, thus preventing unauthorized users from making changes to important files and settings. This also has a downside – users with less than administrator privileges often find themselves unable to install applications without direct assistance from a system administrator.



When the “Always Install with Elevated Privileges” settings is enabled in the system group policy editor, the Windows Installer service permits users with limited access rights to perform system level changes to the system – that is, install files and registry entries into locations where the logged on user would not ordinarily have access. This is possible because Windows Installer is a system level service and runs with elevated privileges.

All installations created with InstallAware automatically take advantage of elevated privileges and successfully install inside many locked-down corporate and other types of secured environments.

Corporate Deployment

One of the main reasons Windows Installer rapidly became a corporate standard is its ease of deployment. If you are managing thousands of workstations on your corporate network, it is simply out of the question to manually install applications on all those workstations. Instead, you have to automate the deployment process. Windows Installer has been adopted rapidly in large organizations because it lends itself very cleanly to such types of mass deployment and maintenance.

Installations created with InstallAware have always supported silent installations, along with a configurable set of command line parameters for specifying advanced installation settings. What's more, in addition to the standard configurability provided by Windows Installer, InstallAware installations can define their own custom command line parameters and give systems administrators even more automated deployment options. Setups execute completely silently in the background and also provide logging options for post-mortem diagnostics in case of complications with the deployment procedure.

Even better, the WebAware Installation technology uniquely featured in InstallAware offers some very exciting opportunities when used in large organizations for deployment. If you specify your web media block sources on corporate file servers (as opposed to Internet URLs), you get several advantages:

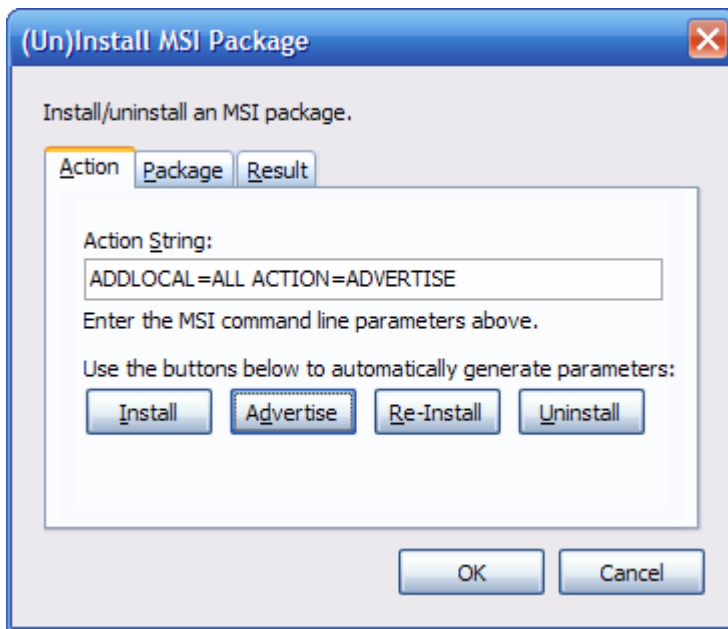
- **Security:** Setups will not run outside of the organization network, as the web media blocks will be accessible only on that network
- **Versioning:** Setups will always have access to the latest version of setup sources, since the centralized web media blocks will always contain the most up to date files
- **Patching:** Especially for patches that require access to source media (a problem that makes patch deployment a nightmare in large environments), web media blocks provide automatic source resolution, even across different versions of the same product

In the true InstallAware tradition, InstallAware 6 takes corporate deployment one step further with the additional features and functionality it offers, on top of those already provided by Windows Installer – saving you time and money with its advanced deployment mechanisms.

Shelling to Other MSI Installations

Windows Installer offers the merge module mechanism for extending your setups. Merge modules are Windows Installer database files with the MSM extension, self contained blocks of installation logic and code, used for installing common parts of applications (such as application runtimes). Merge modules are “merged” into the main installation database, the MSI file, at build time, and with absolutely no effort on part of the developer, extend the MSI database with their additional logic and code, providing for the installation of the application runtimes, and so on.

InstallAware 6 supports merge modules, but again adds an extra bit of functionality. With InstallAware 6, you may (un)install or reconfigure complete third party products, given their MSI files, or product GUIDs, in addition to merging MSM databases with your setup.



The (Un)Install MSI Package command provides unprecedented flexibility in “shelling to” third party MSI packages. You may perform the full range of setup actions that are supported by Windows Installer on shelled packages – install or advertise, select all features or just a partial set, and even modify/remove an existing installation.

The command lets you customize the exact command line that will be used in the shelled-to installation, offers the ability to log setup execution, and reports whether the operation was successful, optionally capturing the error message if it failed. Setup packages are identified by their GUID or full path to the setup database. What's best, InstallAware even captures the progress of the shelled-to installation, displaying it as an integral part of your own application installation. Of course, you are able to completely hide the end-user interface of the MSI package you are shelling-to, including hiding any potentially confusing setup choices and dialogs. You get to install the packages with the exact settings you require, with no effort/confusion on part of the end-user running your master setup.

Shelling to setups provides a very viable alternative to application repackaging. Unlike application repackaging which is error-prone, and loses the intent of the original package author, shelling to setups preserves both the original setup logic of the installation, and adds the benefit of being a completely seamless and integral part of your own, larger installation.

Possible applications of this technology are endless. A very viable use for the shelling technology is resolving application conflicts automatically. For instance, if an application is detected on the end-user system which conflicts with your product, it can automatically be removed, all as a setup pre-requisite for your own installation. The Is MSI Setup Installed command reports whether a particular MSI package has been installed on the system, and complements the shelling facility in resolving application conflicts – all with zero end-user intervention/confusion.

Summary

InstallAware 6 achieves what no other installation development environment can, providing you with state-of-the-art tools that save you time and effort while building modern application installations. InstallAware 6's unique fusion of scripting, web deployment, and Windows Installer technologies makes InstallAware 6 the ultimate setup development solution.

About InstallAware Software Corporation

Focusing solely on the Windows Installer (MSI) platform for Microsoft Windows operating systems, InstallAware is the premier provider of software installation tools for Internet deployment. Founded in 2003, InstallAware is supported by thousands of users worldwide, and is a Borland Technology Partner. For more information, visit www.installaware.com.

Copyright© 1996-2006 InstallAware Software Corporation. All rights reserved. All InstallAware brand and product names are trademarks or registered trademarks of InstallAware Software Corporation in the United States and other countries. Microsoft, Windows, and other Microsoft product names are trademarks or registered trademarks of Microsoft Corporation in the U.S. and other countries. All other marks are the property of their respective owners. Corporate Headquarters: 336 Guerrero Street, San Francisco CA 94103-3332 · 415-358-4094 · www.installaware.com · sales@installaware.com