InstallShield 2013

Release Notes

originally released June 2013; updated to include SP1, released October 2013

Introduction

InstallShield is the industry standard for authoring high-quality Windows Installer– and InstallScript-based installations, as well as Microsoft App-V packages. InstallShield 2013 offers new features and enhancements that make it easy to use the latest technologies.

For the latest information about InstallShield 2013, including updates to these release notes, see Knowledge Base article Q210472.

Changes in SP1 (October 2013)

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

To obtain SP1, see KB article Q213736.

Concurrent License Note: If you are using a concurrent license of InstallShield, you must update the version of the FlexNet Licensing Server software on the licensing server before you can start using InstallShield 2013 SP1. The version of the FlexNet Licensing Server software that shipped with the original release version of InstallShield 2013 cannot manage licenses of InstallShield 2013 SP1.

Note that the new version of the FlexNet Licensing Server software can manage licenses of both InstallShield 2013 SP1 and the original release version of InstallShield 2013.

The installation for this new version of the FlexNet Licensing Server software is available in KB article Q213736.

Ability to Target Windows 8.1 and Windows Server 2012 R2 Systems

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

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

- FSharp Redistributable Package 2.0
- JRE_SE 1.7.0_02 (x64)
- JRE_SE 1.7.0_02 (x86)
- Microsoft .NET Framework 3.0 OS Component
- Microsoft .NET Framework 3.5 SP1 (Windows Feature)

- Microsoft .NET Framework 4.5 Full
- Microsoft .NET Framework 4.5 Web
- Microsoft App-V 5.0 SP1 Desktop Client (x64)
- Microsoft App-V 5.0 SP1 Desktop Client (x86)
- Microsoft ReportViewer 2010
- Microsoft SQL CE 3.5 SP2
- Microsoft SQL Server 2005 Express SP3 (x86 & x64Wow)
- Microsoft SQL Server 2005 Express SP3 (x86)
- Microsoft SQL Server 2008 Express SP1 (x64)
- Microsoft SQL Server 2008 Express SP1 (x86 & x64Wow)
- Microsoft SQL Server 2008 Express SP1 (x86)
- Microsoft SQL Server 2008 Management Objects 10.00.2531 (IA64)
- Microsoft SQL Server 2008 Management Objects 10.00.2531 (x64)
- Microsoft SQL Server 2008 Management Objects 10.00.2531 (x86)
- Microsoft SQL Server 2008 Native Client 10.00.2531 (IA64)
- Microsoft SQL Server 2008 Native Client 10.00.2531 (x64)
- Microsoft SQL Server 2008 Native Client 10.00.2531 (x86)
- Microsoft SQL Server 2008 R2 Express RTM (x64)
- Microsoft SQL Server 2008 R2 Express RTM (x86 & x64Wow)
- Microsoft SQL Server 2008 R2 Express RTM (x86)
- Microsoft SQL Server 2008 R2 Express SP2 (x64)
- Microsoft SQL Server 2008 R2 Express SP2 (x86 & x64Wow)
- Microsoft SQL Server 2008 R2 Express SP2 (x86)
- Microsoft SQL Server 2008 R2 Native Client 10.50.1600.1 (IA64)
- Microsoft SQL Server 2008 R2 Native Client 10.50.1600.1 (x64)
- Microsoft SQL Server 2008 R2 Native Client 10.50.1600.1 (x86)
- Microsoft SQL Server 2012 Express LocalDB RTM (x64)
- Microsoft SQL Server 2012 Express LocalDB RTM (x86)
- Microsoft SQL Server 2012 Express RTM (x64)
- Microsoft SQL Server 2012 Express RTM (x86 & x64Wow)
- Microsoft SQL Server 2012 Express RTM (x86)
- Microsoft SQL Server 2012 Native Client (x64)
- Microsoft SQL Server 2012 Native Client (x86)
- Microsoft SQL Server Compact 4.0 (x64)
- Microsoft SQL Server Compact 4.0 (x86)
- Microsoft SQL Server Native Client 9.00.4035 (IA64)
- Microsoft SQL Server Native Client 9.00.4035 (x64)
- Microsoft SQL Server Native Client 9.00.4035 (x86)

- Microsoft SQL Server System CLR Types 10.00.2531 (IA64)
- Microsoft SQL Server System CLR Types 10.00.2531 (x64)
- Microsoft SQL Server System CLR Types 10.00.2531 (x86)
- Microsoft Visual C++ 2005 SP1 Redistributable MFC Security Update KB2538242(x64)
- Microsoft Visual C++ 2005 SP1 Redistributable MFC Security Update KB2538242(x86)
- Microsoft Visual C++ 2005 SP1 Redistributable Package (x64)
- Microsoft Visual C++ 2005 SP1 Redistributable Package (x86)
- Microsoft Visual C++ 2008 SP1 Redistributable MFC Security Update KB2538243(x64)
- Microsoft Visual C++ 2008 SP1 Redistributable MFC Security Update KB2538243(x86)
- Microsoft Visual C++ 2008 SP1 Redistributable Package (x64)
- Microsoft Visual C++ 2008 SP1 Redistributable Package (x86)
- Microsoft Visual C++ 2010 Redistributable Package (x64)
- Microsoft Visual C++ 2010 Redistributable Package (x86)
- Microsoft Visual C++ 2010 RTM Redistributable MFC Security Update KB2467173 (x64)
- Microsoft Visual C++ 2010 RTM Redistributable MFC Security Update KB2467173 (x86)
- Microsoft Visual C++ 2010 SP1 Redistributable Package (x64)
- Microsoft Visual C++ 2010 SP1 Redistributable Package (x86)
- Microsoft Visual C++ 2012 Redistributable Package (x64)
- Microsoft Visual C++ 2012 Redistributable Package (x86)
- Microsoft Visual C++ 2012 Update 1 Redistributable Package (x64)
- Microsoft Visual C++ 2012 Update 1 Redistributable Package (x86)
- Microsoft VSTO 2010 Runtime (x64)
- Microsoft VSTO 2010 Runtime

Enhancements to the InstallScript Language for Windows 8.1 and Windows Server 2012 R2

The following structure members and predefined constants were added to the InstallScript language:

- **SYSINFO.WINNT.bWin81**—This is a new SYSINFO structure member. If the operating system is Windows 8.1 or Windows Server 2012 R2, this value is TRUE.
- **ISOSL_WIN81**—This is a new predefined constant that is available for use with the FeatureFilterOS function and the SYSINFO structure variable. It indicates that the target system is running Windows 8.1 or Windows Server 2012 R2.

Automation Interface Enhancement: OSFilter Property Value for Windows 8.1 and Windows Server 2012 R2

The following constants are now available for use with the OSFilter member of the ISWiComponent and ISWiRelease objects in the automation interface:

eosWin81 = &H8000000 (134217728)—These are for Windows 8.1 and Windows Server 2012 R2.

In addition, the value for the eosAll constant is now &HFD100D0 (265355472); previously, it was &7D100D0 (131137744).

The OSFilter member applies to the ISWiComponent object in InstallScript and InstallScript MSI projects. The OSFilter member applies to the ISWiRelease object in InstallScript projects.

Support for Microsoft Visual Studio 2013

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

New InstallShield Prerequisites for Microsoft SQL Server 2012 Express SP1

InstallShield includes several new SQL Server–related InstallShield prerequisites that you can add to Basic MSI, InstallScript, and InstallScript MSI projects:

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

These InstallShield prerequisites install the technology on supported target systems.

This functionality resolves the following issues: IOA-000081241 and IOA-000081242.

Support for Defining Custom Themes for the Controls on Wizard Pages and Windows in Suite/Advanced UI and Advanced UI Installations

Suite/Advanced UI and Advanced UI projects now enable you to define themes for wizard interface controls. A control theme is a collection of colors for various parts of controls—text color, background color, and border color—and for various states of controls—such as clicked and hovered. You can specify the control theme that you want to use for the controls in your user interface by default. You can also override the theme for specific controls on wizard pages and windows in your user interface. Control themes give buttons and other items in your installation's user interface a flat look (without the three-dimensional effects) that is reminiscent of the style of Windows Store apps.

The following types of controls support the use of control themes:

- Buttons, including the navigation buttons
- Check boxes
- Radio buttons
- Command links

To add a control theme to your project, right-click the Control Themes node in the Wizard Interface view, and then click Add Control Theme. Then configure the theme's settings as needed.

To specify the theme that you want to use by default for the controls in your user interface, select the appropriate theme in the new Default Control Theme setting that is displayed in the Wizard Interface view when the Wizard Pages node is selected. This setting is blank by default.

To override the control theme for a specific control, select the appropriate theme in the Control Theme setting for that control.

New Predefined Path Variable for the Visual Studio Solution Folder

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

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

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

- -6103: Could not find file <VSSolutionFolder>\MyFile.exe
- -6271: File <VSSolutionFolder>\MyFile.exe not found. An error occurred building the MsiFileHash table record for this file. Verify that the file exists in the specified location.

This functionality is available in the following project types: Advanced UI, Basic MSI, InstallScript, InstallScript MSI, Merge Module, Suite/Advanced UI.

Ability to Easily Update the Product Code of All Instances in a Multi-Instance Project

If you are creating major upgrades for a project that includes multi-instance support, it is necessary to update the product code of each instance. InstallShield now enables you to easily update the product code of a specific instance that is defined in your project. InstallShield also now lets you easily update the product code of all instances that are defined for a product configuration in your project.

To update the product code of a specific instance, use the new **Generate a new GUID** command that is available when you right-click the value in the ProductCode setting for an instance that is defined on the Multiple Instances tab for a product configuration in the Releases view.

To update the product code of all of the instances that are defined for a product configuration in your project, use the new Change ProductCode on All Instances command that is available when you right-click the Instances node on the Multiple Instances tab for a product configuration in the Releases view.

This functionality is available in Basic MSI projects.

IOA-000059192 (Basic MSI)

When the InstallShield prerequisites for Windows Installer are included in an installation, the InstallWelcome dialog is no longer displayed behind the other installation-related dialogs.

IOA-000070914 (Advanced UI, Suite/Advanced UI)

InstallShield no longer crashes at build time while building an Advanced UI or Suite/Advanced UI installation that contains more than one dynamic link that points to the same folder.

IOA-000074927 (Basic MSI, DIM, InstallScript MSI, Merge Module)

If you create a "Folder path, by searching in a specific folder" type of system search in your project, InstallShield no longer changes it to a different type of system search.

IOA-000078306 (Advanced UI, Suite/Advanced UI)

If an end user runs an Advanced UI or Suite/Advanced UI installation or uninstallation that is configured to remove an .exe package from the target system, and the end user cancels the uninstallation of the .exe package through its own external user interface, the Advanced UI or Suite/Advanced no longer indicates that the uninstallation was successful. Previously, the last wizard page in the Advanced UI or Suite/Advanced UI indicated that the process was successful, even though the .exe package was not removed.

IOA-000079636 (InstallScript)

If you specify a valid custom icon for the Setup.exe file of an InstallScript installation, InstallShield is now able to use the custom icon instead of the default icon. In addition, InstallShield no longer generates build warning -5031, explaining that it could not use the custom icon.

IOA-000079780 (Basic MSI, DIM, InstallScript, InstallScript MSI, Merge Module)

The Registry view has been optimized to better handle large numbers of registry entries, which is seen primarily in projects that were captured by AdminStudio Repackager.

IOA-000080073 (InstallScript MSI)

The SQLLogin dialog in an InstallScript MSI installation now displays the default values that were configured in the SQL Scripts view of the project. Previously, the fields on this dialog were blank at run time.

IOA-000080471

If the path that is passed to the InstallScript function GetDir ends with a path separator, GetDir now returns the appropriate path. Previously, the trailing path separator caused GetDir to fail, unless the function StrRemoveLastSlash was used to remove the separator before the GetDir call.

IOA-000080478 (Transform)

It is now possible to change the Attributes field for an entry in the Upgrade table in the Direct Editor view of a transform project without InstallShield crashing.

IOA-000080479 (InstallScript)

If the InstallScript function FeatureAddItem is called during an InstallScript installation to add subfeatures to a script-created feature set, a call to SdFeatureTree, SdFeatureDialog2, or SdFeatureMult no longer results in a run-time crash.

IOA-000080559 (Basic MSI, DIM, InstallScript MSI, Merge Module)

If an end user runs repair for an installation that added to the target system a scheduled task that was configured in the Scheduled Tasks view, the repair operation no longer removes the scheduled task.

IOA-000080707

The Japanese version of InstallShield no longer displays garbled text for elements such as menu commands and toolbars when InstallShield is used from within Visual Studio.

IOA-000080843 (InstallScript, InstallScript MSI)

If an .exe that is called by the InstallScript function DoInstall fails, the installation returns an error code and proceeds without crashing. Previously in this scenario, the installation crashed.

IOA-000081120

It is now possible to save changes to a patch creation properties file (.pcp) when it is open in InstallShield in direct edit mode.

IOA-000081302 (Basic MSI, InstallScript MSI)

InstallScript custom action code that uses the abort statement now triggers the installation to abort under appropriate conditions as expected. Previously, the custom action that contained the abort statement returned a value that indicated an error instead of the end user aborting. Thus, the abort statement may not have worked. In one scenario on Windows 8.1 target systems, the issue caused the custom action to stop running but the installation continued.

Note that the abort statement will now result in a custom action returning ERROR_INSTALL_USEREXIT (1602). This does not abort an installation when called from a control event. To abort in that case, your custom action must instead return ERROR_INSTALL_FAILURE (1603).

IOA-000081442 (Basic MSI, InstallScript MSI)

If an installation uses the built-in SQL script support and more than one SQL Server instance is installed locally on the target system, the SQLBrowse dialog no longer shows garbage characters in place of local instance names.

IOA-000081492

If you use InstallShield from within a Visual Studio solution and you add a new XML file to the XML File Changes view, InstallShield no longer crashes. Previously in this scenario, if the InstallShield project did not contain any features or components, Visual Studio crashed when the XML file was added in the XML File Changes view.

IOA-000081758 (InstallScript, InstallScript MSI)

If an end user clicks a hyperlink in an HTML control, a message indicating that the link click occurred is now sent to the WaitOnDialog loop. Previously, this message was not sent, so InstallScript code that depended on notification of the click event may have behaved unexpectedly.

IOA-000081790 (Basic MSI, InstallScript MSI)

If an end user specifies a new unused account for the /user: command-line parameter when using the runas Windows command while launching a Windows Installer package that has one or more InstallScript custom actions, the InstallScript custom actions no longer trigger a 1603 run-time error.

IOA-000081955 (InstallScript, InstallScript MSI)

If the InstallScript function CtrlSetText is used to change the HTML files that an HTML control displays at run time, CtrlSetText now displays the correct HTML files. Previously under certain conditions, the first call to CtrlSetText displayed the correct HTML file, but subsequent calls failed and returned ISERR_GEN_FAILURE.

New Features in InstallShield 2013 Original Release Version (June 2013)

Ability to Create Run-Time Actions That Extend the Behavior of a Suite/Advanced UI Installation

You may require your Suite/Advanced UI installation to perform various run-time tasks that are outside the scope of the packages that you are including in the installation. For example, you may need the Suite/Advanced UI installation to do one or more of the following:

- Install an application before the user interface of the Suite/Advanced UI installation is displayed. One sample scenario in which this may be necessary is when one of the packages in your Suite/Advanced UI installation runs SQL scripts to install an Oracle database. Before this package is run, you may want the Suite/Advanced UI interface to display a wizard page that lets end users select the appropriate server from a list of servers that are available on the network. This type of UI support requires that ODBC drivers be installed on the target system.
- Search the target system for the presence or absence of a particular product, technology, folder, file, registry entry, or other item.
- Configure the target system before or after running a package in the Suite/Advanced UI installation.

To extend the capabilities of a Suite/Advanced UI installation and make it possible to perform those sorts of tasks and more, you can use the new Events view in your Suite/Advanced UI project to create actions that run executable files, call DLL functions, run PowerShell scripts, or set Suite/Advanced UI properties at run time.

When you add an action to your project, you can schedule when at run time you want it be launched:

- Use the Events explorer in the new Events view to schedule the action to run during one or more of the built-in events that the Suite/Advanced UI manages. This area of the view lists the events and the actions that occur during each event in chronological order from top to bottom.
- Use the settings in the new Events area of the Packages view to schedule an action to run before or after the Suite/Advanced UI engine installs, removes, modifies, or repairs a package.

When you schedule an action, you can build conditional statements that control whether the action should be run. You can use the same types of condition checks that are available in other areas of a Suite/Advanced UI project, as well as two additional new types of condition checks:

- **Package Operation**—Check the state (for example, install or remove) in which a particular package in the Suite/Advanced UI installation is running.
- **Feature Operation**—Check the state (for example, install or remove) in which a particular feature in the Suite/Advanced UI installation is running.

These types of conditions are available only for actions that are associated with an event in the Events view, or with a package in the Packages view.

Note that the PowerShell action support requires PowerShell 2.0 or later on target systems.

This new feature is available in the Premier edition of InstallShield.

This feature resolves issue IOA-000069026.

Support for Enabling Windows Roles and Features During a Suite/Advanced UI Installation

If a particular package in your Suite/Advanced UI installation requires that one or more Windows roles and features be enabled on target systems, you can specify this requirement through the new Windows Features setting in the Packages view when you are configuring the package in your Suite/Advanced UI project. At run time, if an eligible

package that is being installed requires one or more Windows roles or features that are disabled, the Suite/Advanced UI installation enables those roles and features before launching the package.

For example, your Suite/Advanced UI project may have a package that installs an IIS Web site that requires that the Internet Information Services feature in Windows be turned on. Another package in the same project may require that the PowerShell feature be turned on. When you are configuring the settings of those packages in your project, you can specify the Windows features that are required; at run time, if a package is eligible to be installed on a target system but any of its required Windows features are disabled, the Suite/Advanced UI installation enables those disabled Windows features before launching it. If the package is not eligible, its required Windows features are not enabled.

InstallShield has built-in support for enabling several Windows features:

- Internet Information Services
- PowerShell
- .NET Framework 3.x

InstallShield also lets you specify additional Windows roles and features that a package requires.

This new feature is available in the Premier edition of InstallShield.

This feature resolves issue IOB-000061772.

Ability to Create Advanced UI and Suite/Advanced UI Project Templates

InstallShield has support for using Advanced UI and Suite/Advanced UI projects as templates. A project template contains default settings and design elements that you want to use as a starting point when you create a new Advanced UI or Suite/Advanced UI project.

You can designate any Advanced UI or Suite Advanced UI project to be a template. Project files and template files have the same file extension: .issuite.

To designate that an .issuite project file is a template file and make the template available for selection in the New Project dialog box, right-click in the All Types tab on the New Project dialog box and then click Add New Template. InstallShield lets you browse to the .issuite file that you want to use as a template, and it adds a new icon for it to the All Types tab.

To create a new Advanced UI or Suite/Advanced UI project using your .issuite file as a template, select the template's icon on the All Types tab of the New Project dialog box when you are creating the new project.

New Built-in Template Available for Creating Installations that Install Multi-tier Applications

InstallShield includes a new Multi-tier Application template that helps you get started with creating a Suite/Advanced UI installation for cloud-based multi-tier applications. You can create this sort of installation to enable end users to easily manage the installation of tiers that install Web sites, databases, and applications. They can use the wizard pages in the Suite/Advanced UI installation to select which tiers they want to install, or they can use the command-line support to select the appropriate features.

The new Multi-tier Application template contains three tiers that are exposed as features. You can customize these features to include the tiers of your product, adding or removing features as needed. The template also contains four releases in the Releases view: one flagged for each of the three predefined features, and one that includes all of the features. This enables you to build separate installations—one for each tier, and one that lets end users install all available tiers.

This feature is available in the Premier edition of InstallShield.

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

InstallShield includes the following InstallShield prerequisites that you can add to Advanced UI, Basic MSI, InstallScript, InstallScript MSI, and Suite/Advanced UI projects:

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

These prerequisites install the various technologies on supported target systems.

This feature resolves issue IOA-000075613.

New Application Virtualization Suitability Tests

Several new validation suites are available in InstallShield for helping you to determine how ready your products are for virtualization. The InstallShield virtualization internal consistency evaluators (ISVICEs) that are included in these suites let you check suitability for Microsoft App-V, Microsoft Server App-V, VMware ThinApp, and Citrix XenApp. The validation suites can enable you to make more informed decisions about how you should build your product if you are considering offering your customers a virtualized version.

If you want to configure InstallShield to perform validation with these validation suites each time that a Basic MSI release is successfully built: On the Tools menu, click Options. On the Validation tab, select the appropriate check boxes.

If you want to perform validation separately from the build process: On the Build menu, point to Validation, and then click the appropriate new suite.

The virtualization suites are available in the Virtualization Pack.

This feature is available in the following project types: Basic MSI and InstallScript MSI.

Support for Creating Microsoft App-V 5.x Packages

The Microsoft App-V Assistant in InstallShield includes support for creating virtualized applications in the Microsoft App-V 5.x format. The Package Information page in this assistant lets you specify which version of App-V—5.x or 4.x—you want to target.

If you are targeting App-V 5.x, a new version of the App-V Launcher tool is available for testing App-V 5.x packages before moving them to the deployment server. The new version of this tool is capable of launching a Command Prompt window within the virtual environment of the App-V package. Thus, it is no longer necessary to inject diagnostic tool shortcuts for Cmd.exe or Regedit.exe directly into an App-V package beginning with App-V 5.x.

The name of the directory in which InstallShield saves an App-V package at build time varies, depending on which version of App-V you are targeting. For App-V 5.x packages, the directory is ProductName, and it is placed in a folder called App-VPackage. For App-V 4.x packages, InstallShield includes the product version number to the ProductName folder; that is, ProductName_v*N*, (where *N* is the product version number).

Note that although you can configure settings for an App-V 5.x package on any version of Windows that InstallShield supports, building an App-V 5.x package from within InstallShield requires Windows 8 or Windows Server 2012. Attempting to build an App-V 5.x package on an earlier version of Windows causes build error -9424.

The Microsoft App-V Assistant is available in the Virtualization Pack.

New Property Comparison Type of Condition Check; New Property for Determining the Install Mode for Suite/Advanced UI and Advanced UI Projects

When you are building a conditional statement for an exit, detection, eligibility, or feature condition in a Suite/Advanced UI or Advanced UI project, or for an action in a Suite/Advanced UI project, you can select from a number of different types of checks that you want to be evaluated on target systems. Use the new Property Comparison type of condition check to check the value of a particular built-in Advanced UI or Suite/Advanced UI property, or a property that is defined in the Property Manager view.

In addition, a new read-only property called ISInstallMode is now available. This property stores a value that indicates the mode (first-time installation, maintenance/UI maintenance mode selection, modify, remove, repair, or stage only) in which the Suite/Advanced UI or Advanced UI installation is running. You can use this property in conditional statements that you configure in your project.

Ability to Build Pure 64-Bit .msi and .msm Packages; New Build-Time Architecture Validation

Windows Server Core supports disabling 32-bit Windows-on-Windows (WOW64) support. As this configuration becomes more popular, you may want to ensure that your 64-bit applications can install without any reliance on 32-bit functionality. To make this possible, InstallShield now enables you to build pure 64-bit .msi packages; these can be run on 64-bit Windows-based systems that do not have WOW64 functionality. You can also build pure 64-bit merge modules and include them in InstallShield projects that have 64-bit support. If your installation or merge module project targets a pure 64-bit environment and includes support that requires any built-in InstallShield custom action DLLs, InstallShield includes new 64-bit versions of these DLLs in your releases at build time.

InstallShield now has architecture validation that you can use when building a release in InstallShield. Architecture validation enables you to detect potentially problematic cases in which your installation may try to install product files or use run-time binaries that may not match the architecture of a target system.

For example, if end users may run your installation on 64-bit target systems that do not have WOW64 support, architecture validation can help you identify any 32-bit product files or 32-bit custom action files in your installation. In addition, if you are mixing 64-bit and 32-bit product files (or 64-bit and 32-bit custom actions) in a single project and generating separate 32-bit and 64-bit .msi packages, you can use architecture validation to identify any 64-bit product or custom action files in your 32-bit releases, since these files cannot be loaded on 32-bit target systems.

To specify what type of architecture validation you want to use and identify whether you want to build a pure 32-bit or 64-bit package, use the new Architecture Validation setting, which is available on the General tab when you select a product configuration in the Releases view. Two types of validation are available: strict and lenient.

Strict validation may trigger build errors if the architecture that the Template Summary property specifies does not match the architecture for one or more of the custom action files that are being included in the release. This type of validation may also trigger build warnings if the architecture that the Template Summary property specifies does not match the architecture for one or more of the product files that are being included in the release.

Lenient architecture validation, which is the default type of validation, does not trigger build errors or warnings if the architecture that the Template Summary property specifies does not match the architecture for one or more of the product files or the custom action files that are being included in the release.

The automation interface includes support for the new architecture validation support. The ISWiProductConfig object includes a new read-write ArchitectureValidation property that lets you specify the type of architecture validation that you want to use for a product configuration.

This feature is available in the following project types: Basic MSI and Merge Module.

This feature resolves issue IOA-000074874.

Validation for the Windows 8 Logo Program

InstallShield includes two new validation suites: InstallShield Validation Suite for Windows 8 and InstallShield Merge Module Validation Suite for Windows 8. These validation suites contain several new InstallShield internal consistency evaluators (ISICEs): ISICE21 through ISICE26. The suites can help you verify whether your Windows Installer–based installation or merge module meets the installation requirements of the Windows 8 Desktop App Certification Program. If you want to be able to use the Windows 8 logo artwork, your product's installation must meet the program's requirements. These suites can also help you verify that your installation or merge module will work on Windows Server 2012 systems.

If you want to configure InstallShield to perform validation with these validation suites each time that a release is successfully built: On the Tools menu, click Options. On the Validation tab, select the appropriate check boxes.

If you want to perform validation separately from the build process: On the Build menu, point to Validation, and then click the appropriate new suite.

Improvements for Configuring Shortcuts on the Latest Platforms; InstallScript Language Improvements for Shortcut Creation

InstallShield offers improvements for configuring shortcuts in your projects.

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

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

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

This feature is available for the following project types: Basic MSI, DIM, InstallScript, InstallScript MSI, Merge Module, MSI Database, MSM Database, and Transform.

The automation interface includes support for specifying whether you want a shortcut to be pinned to the Windows 8 Start screen. The ISWiShortcut object includes a new read-write Boolean EnableWin8StartPin property that indicates whether the shortcut is configured to be pinned by default to the Start screen on Windows 8 target systems.

Support for Preventing End Users from Pinning a Shortcut to the Taskbar or Start Menu

For each shortcut in your installation, InstallShield lets you specify whether you want to let end users pin the shortcut to the taskbar or to the Start menu. If you want to prevent this type of pinning, the installation sets a property that hides the context menu commands for pinning the shortcut to the taskbar and the Start menu. You may want to prevent pinning for shortcuts that are for tools and secondary products that are part of your installation.

Support for preventing end users from pinning a shortcut to the taskbar or Start menu varies, depending on the project type that you are using.

In Basic MSI, DIM, InstallScript MSI, Merge Module, MSI Database, MSM Database, and Transform projects: To prevent taskbar and Start menu pinning for a shortcut, use the Shell Properties setting for a shortcut in the

Shortcuts view. This setting has a new Prevent Pinning option that lets you disable pinning for the selected shortcut. Windows Installer 5 includes support for this shortcut setting. Previously, it was necessary to manually enter the appropriate property name and value to configure this behavior.

In InstallScript projects: To prevent taskbar and Start menu pinning for a shortcut, use the new Prevent Pinning setting in the Shortcuts view. Windows 7 and later include support for this shortcut setting. Previously, InstallShield did not have support for configuring this functionality in InstallScript projects.

The automation interface includes support for specifying whether you want the context menu commands for pinning a shortcut to the taskbar and to the Start menu to be displayed after end users install your product. The ISWiShortcut object includes a new read-write Boolean PreventPinning property that indicates whether the shortcut is configured to hide the context menu commands for pinning.

Support for Preventing a Shortcut on the Start Menu from Being Highlighted as Newly Installed

InstallShield lets you optionally prevent the Start menu entry for a shortcut from being highlighted as newly installed after end users install your product. This has the same effect as clearing the **Highlight newly installed programs** check box in the Customize Start Menu dialog box for an individual item on a target system. You may want to set this property for shortcuts that are for tools and secondary products that are part of your installation.

Support for the highlighting behavior varies, depending on the project type that you are using.

In Basic MSI, DIM, InstallScript MSI, Merge Module, MSI Database, MSM Database, and Transform projects: To prevent the highlighting behavior for a shortcut, use the Shell Properties setting for a shortcut in the Shortcuts view. This setting has a new Do Not Highlight as New option that lets you disable highlighting for the selected shortcut. Windows Installer 5 includes support for this shortcut setting. Previously, it was necessary to manually enter the appropriate property name and value to configure this behavior.

In InstallScript projects: To prevent the highlighting behavior for a shortcut, use the new Do Not Highlight as New setting in the Shortcuts view. Windows 7 and later include support for this shortcut setting. Previously, InstallShield did not have support for configuring this functionality in InstallScript projects.

The automation interface includes support for specifying whether you want a shortcut to be highlighted on the Start menu as new after end users install the product. The ISWiShortcut object includes a new read-write Boolean DoNotHighlightAsNew property that indicates whether you want to prevent the highlighting.

Built-in Support for Configuring Additional Windows Shell Properties for a Shortcut

InstallShield lets you specify one or more additional shortcut properties that need to be set by the Windows Shell at run time. The properties that the Shell can set are defined in propkey.h, which is part of the Windows SDK.

To configure additional properties for a shortcut, use the Shell Properties setting for a shortcut in the Shortcuts view. This setting has a new Custom Property option that lets you specify additional properties and their corresponding values for the selected shortcut.

Windows Installer 5 includes support for configuring Shell properties.

The automation interface includes a new ISWiShellProperty object for custom Windows Shell properties for a shortcut. In addition, the ISWiShortcut object includes two new methods and a collection that let you add a Shell property (AddShellProperty), delete a Shell property (DeleteShellProperty), and retrieve the collection of Shell properties (ISWiShellProperties) for a shortcut.

This support is available in the following project types: Basic MSI, DIM, InstallScript MSI, Merge Module, MSI Database, MSM Database, and Transform.

InstallScript Language Improvements for Setting a Shortcut Property, and for Other Shortcut Functionality

A new InstallScript function called SetShortcutProperty is available. You can use this function in your InstallScript code to set Windows Shell properties for a shortcut. The function lets you prevent end users from pinning a shortcut to the Windows 8 Start screen, prevent end users from pinning a shortcut to the taskbar or Start menu on Windows 7 or later systems, or prevent a shortcut on the Start menu from being highlighted as newly installed on Windows 7 or later systems. You can also use this function to specify additional Shell properties that are defined in propkey.h, which is part of the Windows SDK.

The InstallShield Cabinet and Log File Viewer has been updated to reflect the state of the aforementioned Shell properties.

To reflect the current operating system terminology, some of the existing shortcut functions have been renamed. Note that the old function names will continue to compile and run as they did with earlier versions of InstallShield. The new InstallScript functions are:

- **CreateShortcut**—This function supersedes AddFolderIcon. CreateShortcut accepts the same parameters as AddFolderIcon but adds or replaces the following options for the nFlag parameter:
  - CS_OPTION_FLAG_REPLACE_EXISTING
  - CS_OPTION_FLAG_RUN_MAXIMIZED
  - CS_OPTION_FLAG_RUN_MINIMIZED
  - CS_OPTION_FLAG_PREVENT_PINNING
  - CS_OPTION_FLAG_NO_NEW_INSTALL_HIGHLIGHT
  - CS_OPTION_FLAG_NO_STARTSCREEN_PIN
- **CreateShortcutFolder**—This function supersedes CreateProgramFolder. Both functions use the same parameters.
- **DeleteShortcut**—This function supersedes DeleteFolderIcon. Both functions use the same parameters.
- **DeleteShortcutFolder**—This function supersedes DeleteProgramFolder. Both functions use the same parameters.
- **GetShortcutInfo**—This function supersedes QueryProgItem. Both functions use the same parameters.
- **ReplaceShortcut**—This function supersedes ReplaceFolderIcon. ReplaceShortcut accepts the same parameters as ReplaceFolderIcon but adds or replaces the same options for the nFlag parameter as CreateShortcut.

This feature resolves the following issues: IOA-000074185, IOA-000074823.

Usability Improvements for Customizing the Wizard Interface in Advanced UI and Suite/Advanced UI Projects

No More Confusing Syntax Requirements

Various UI settings in the Wizard Interface view in Advanced UI and Suite/Advanced UI projects have been improved to make it much easier to customize the wizard interface of Advanced UI and Suite/Advanced UI installations. For example, the following settings, which are displayed when a wizard page or wizard control is selected in the Wizard Interface view, have been improved:

- **Visible**—This setting lets you specify one or more conditions that the Advanced UI or Suite/Advanced UI installation should use to evaluate whether the selected page or control should be displayed.

- **Enabled**—This setting lets you specify one or more conditions that the Advanced UI or Suite/Advanced UI installation should use to evaluate whether the selected control should be enabled.
- **Validate**—This setting lets you specify one or more validation statements that the Advanced UI or Suite/Advanced UI installation should use to validate an end user's response to a control. For example, you can use this setting to specify the format that end users should use for entering a serial number in a text box control.
- **Action**—Each instance of this setting has been renamed to better reflect the specific type of trigger that is applicable to the selected control. For example, the Action setting for a list box control has been renamed Selection Changed. The Action setting for a button has been renamed Click. The renamed settings let you select one or more actions that you want to be triggered when an end user uses the selected control.

Now those settings contain drop-down lists and buttons that are similar to the ones that are available in other areas of Advanced UI and Suite/Advanced UI projects for quickly building conditional statements. These settings now make it easy to quickly configure visible/hidden status, enabled/disabled status, validation, and action responses for various parts of the UI, and to build conditional statements for that behavior if appropriate. Previously, these settings were edit fields that required manually entry of statements using a difficult-to-remember syntax.

Built-In Support for Using Only One Background for the Header, Body, and Navigation Areas of the Wizard Interface

The Wizard Pages node in the Wizard Interface view has a new Full Wizard Background setting. This setting lets you specify a single background that you want to use across the header, body, and navigation areas of your wizard pages. Previously, the only built-in way to specify backgrounds in InstallShield was to use separate background styles in the Default Body Background, Header Background, and Navigation Background settings. If you wanted to use a single background across all three areas of the wizard interface, you had to use the process that was documented in an InstallTalk blog article; the process involved modifying the InstallShield project file (.issuite) in a text editor.

Note that if you select a background in this new Full Wizard Background setting, you should delete any values from the other background settings: Default Body Background, Header Background, and Navigation Background.

Ability to Specify a Global Text Style for All Navigation Buttons

The Wizard Pages node in the Wizard Interface view has a new Navigation Text Style setting. This setting lets you select a single text style that you want to use for the text on all of the navigation buttons. You can override the global text style for individual navigation buttons as needed.

Logically Organized Setting Grids

In the Wizard Interface view, the setting grids for wizard pages, secondary windows, and wizard controls have been reorganized to make it easier to find the settings that are needed for customizing the wizard interface.

This feature resolves the following issues: IOA-000070547, IOA-000075915, IOA-000076012.

Support for Defining Font Sets with Optional Language-Specific Choices for Advanced UI and Suite/Advanced UI Projects

Advanced UI and Suite/Advanced UI projects have support for a new kind of wizard interface style: a font set. A font set is a collection of fonts (including attributes such as font name, size, and weight). For each font in a font set, you can specify to which language the font is applicable. This enables you to select a different font for each language that your project supports.

Font sets work in conjunction with text styles. Text styles, which are styles that define text attributes such as text color and alignment, now reference a font set but can optionally override various font attributes that are defined at the font set level.

Thus, for example, when you are specifying font information for the body text of your wizard interface, you can use a font set called BodyFonts, and specify a different font for each of the languages that your project supports. You can then select the BodyFonts font set for various text styles—Body, Header, and BodyBold—but with special overrides for smaller sizes and bold where necessary. This enables you to specify a large set of possible fonts once, and change attributes such as text size or color for specific wizard interface uses.

Enhancements in InstallShield 2013 Original Release Version (June 2013)

New Services View

InstallShield has a new Services view—under the System Configuration node of the View List—that lets you specify the required information about a Windows service that you are installing, starting, configuring, stopping, or deleting at run time.

This new Services view has been added to make it easier for new users to configure services. The view provides the same functionality as the following existing areas:

- The Services area under the Advanced Settings node for a component in the Setup Design view (of installation projects)
- The Services area under the Advanced Settings node for a component in the Components view

If you configure service information in any of the three views (the Services view, the Setup Design view, or the Components view), the other views are updated automatically.

The Services view is available in the following project types: Basic MSI, DIM, InstallScript MSI, Merge Module, MSI Database, MSM Database, and Transform.

New Wizard Format for the Suite/Advanced UI and Advanced UI Wizard Interface

A new Glass wizard format is available for the wizard interface of Suite/Advanced UI and Advanced UI installations. This new format is similar to the existing Aero format. For both formats, the installation uses user-chosen system colors (instead of brush styles that you defined in your project) to display the caption bar, header, and navigation areas of the wizard interface. It also may use the glass effect (translucency) for these same areas of the wizard interface on target systems that have the translucency support that was introduced in Windows Vista.

In some scenarios, an installation uses the Wizard 97 format instead of the Glass or Aero formats:

- Windows 8 and Windows Server 2012 systems do not have translucency support. On these systems, a Glass-formatted wizard interface uses the Wizard 97 format. However, an Aero-formatted wizard interface uses Aero (which includes user-chosen system colors), but without the glass effect.
- Windows 7, Windows Vista, Windows Server 2008 R2, and Windows Server 2008 have translucency support that end users can enable or disable. On these systems, Glass-formatted wizard interfaces and Aero-formatted wizard interfaces use the Wizard 97 format if translucency is disabled. These wizard interfaces also use the Wizard 97 format if the desktop composition feature on the target system is disabled.
- On Windows XP and Windows Server 2003, Glass-formatted wizard interfaces and Aero-formatted wizard interfaces use the Wizard 97 format.

To configure the format, use the Wizard Format setting that is displayed in the Wizard Interface view when the Wizard Pages node is selected. The Glass format is the default option for all new Suite/Advanced UI and Advanced UI projects.

This enhancement resolves issue IOA-000074604.

Automation Interface Enhancement: New OnUpgrade Property for Minor Upgrades and Small Updates

A new read-write OnUpgrade property has been added to the ISWiProject object in the automation interface. You can set this property to one of the supported values to specify whether you want the installation to display a prompt before installing a minor upgrade or a small update.

This property corresponds with the Small/Minor Upgrade Settings options on the Common tab in the Upgrades view.

This enhancement resolves issue IOA-000077785.

Important Information

Evaluating InstallShield

If you have not purchased a license for InstallShield, you can install it and use it for a limited number of days without activating it or connecting it to a license server. When you use InstallShield before activating it or connecting it to a license server, it operates in evaluation mode, and some of its functionality is not available. For details, see KB article Q200900. Note that the evaluation limitations are removed when you activate InstallShield or when you connect it to a license server and check out a license for it.

Concurrent Licensing Changes

If you purchase concurrent licenses of InstallShield, InstallShield Collaboration—also known as InstallShield Developer Installation Manifest (DIM) Editor—or the Standalone Build, a new version of the FlexNet Licensing Server software is required for the licensing server that you set up to manage your organization's licenses. This software enables users to borrow a license for InstallShield or InstallShield Collaboration from the FlexNet Licensing Server for a specified number of days. Using a borrowed license enables you to use the product while being disconnected from the same network as the FlexNet Licensing Server.

The new version of the FlexNet Licensing Server software requires you to activate the licenses on the licensing server using an activation code, which is available through the Flexera Software Product and License Center. For instructions on how to set up the server, see the InstallShield download and licensing instructions.

Obtaining the Installations for InstallShield, InstallShield Add-Ons, and the Redistributable Files

You can obtain the installations of InstallShield and the Standalone Build through either of the following methods:

- If you have the InstallShield DVD, the installations are on the DVD and you can find them using the DVD Browser.
- The InstallShield and Standalone Build installations are available for download as documented in the InstallShield download and licensing instructions.

Additional installations—such as the redistributable files for the InstallShield prerequisites that are included in InstallShield, the .NET language pack prerequisite files (.prq), and InstallScript objects—are also available in those same locations.

The ability to create DIM projects is available in the Premier edition of InstallShield. This support is also available in the InstallShield Developer Installation Manifest (DIM) Editor. The DIM Editor is included on the InstallShield Premier DVD. It is also available for download from the same location as InstallShield and the Standalone Build.

Installing More than One Edition of InstallShield

Only one edition of InstallShield 2013—Premier, Professional, or Express—can be installed on a system at a time. In addition, the InstallShield 2013 DIM Editor cannot be installed on the same machine with any edition of InstallShield 2013.

Microsoft Visual Studio can be integrated with only one version of InstallShield at a time. The last version of InstallShield that is installed or repaired on a system is the one that is used for Visual Studio integration.

Installing More than One Version of InstallShield

InstallShield 2013 can coexist on the same machine with other versions of InstallShield.

The InstallShield 2013 Standalone Build can coexist on the same machine with other versions of the Standalone Build. In most cases, the Standalone Build is not installed on the same machine where InstallShield is installed. If you do install both on the same machine and you want to use the automation interface, review the "Installing the Standalone Build and InstallShield on the Same Machine" help topic in the InstallShield Help Library to learn about special registration and uninstallation considerations.

InstallShield MSI Log Analyzer

The InstallShield MSI Log Analyzer is no longer supported.

Project Upgrade Alerts

The following information describes possible upgrade issues that may occur when you upgrade projects that were created with InstallShield 2012 Spring and earlier to InstallShield 2013. It also alerts you to possible changes in behavior that you may notice between new InstallShield 2013 projects and projects that are upgraded from InstallShield 2012 Spring or earlier to InstallShield 2013. For updates to this information, see Knowledge Base article Q210474.

General Information about Upgrading Projects that Were Created in Earlier Versions of InstallShield

If you use InstallShield 2013 to open an project that was created with an earlier version, InstallShield 2013 displays a message box that asks you if you want to convert the project to the new version. If you reply that you do want to convert it, InstallShield creates a backup copy of the project with a file extension such as .773 (for an .ism project) or .2012.4 (for an .issuite project) before converting it. Delete the .773 or .2012.4 part from the original project's file name if you want to reopen the project in the earlier version of InstallShield. Note that you cannot open InstallShield 2013 projects in earlier versions of InstallShield.

You can upgrade projects that were created with the following versions of InstallShield to InstallShield 2013: InstallShield 2012 Spring and earlier, InstallShield 12 and earlier, InstallShield DevStudio, InstallShield Professional 7 and earlier, and InstallShield Developer 8 and earlier. Note that projects that were created with InstallShield MultiPlatform or InstallShield Universal cannot be upgraded to InstallShield 2013.

Change in Requirements for Target Systems

InstallShield no longer supports the creation of installations for Windows 2000 systems. If end users have Windows 2000 on their machine and they try to run an installation that was built with InstallShield 2013, the installation

may run successfully. Or, unexpected results may occur, unless the project includes launch conditions that prevent end users from running the installation on this legacy operating system.

For Basic MSI and InstallScript MSI projects, you may want to consider adding a launch condition that displays a message if end users are running your installation on a Windows 2000 system. For InstallScript projects, you may want to consider adding InstallScript code that uses the SYSINFO structure variable to check for Windows 2000 systems, and then display a message if appropriate.

Advanced UI and Suite/Advanced UI installations now require at least Windows XP SP3 or Windows Server 2003 SP2.

InstallShield no longer supports the creation of installations for mobile devices. Thus, the Mobile Devices view, the Smart Device project type, the Palm OS Object, and the Windows Mobile Object are no longer included in InstallShield. If you try to upgrade a Smart Device project from InstallShield 2012 Spring or earlier to InstallShield 2013, InstallShield 2013 displays an error message and fails to open the project. If you upgrade a project from InstallShield 2012 Spring or earlier to InstallShield 2013, and if the project targets desktop platforms and contains other mobile device support, InstallShield removes the mobile device support during the upgrade and logs a warning.

Changes for the Virtualization Pack

The Virtualization Pack is now available as part of the standalone version of InstallShield Premier Edition. It is no longer available as an add-on for the Professional edition of InstallShield.

Repackager Project Conversion Tool No Longer Included

The Repackager is no longer included in the Premier edition of InstallShield. The Repackager is now available only with AdminStudio.

Removal of Outdated and Obsolete Items

InstallShield no longer has support for a number of outdated technologies. Many of these technologies have reached their end-of-life milestone and are no longer supported by Microsoft. For a full list of items, see Knowledge Base article Q212461.

Changes to the XML Schema for Advanced UI and Suite/Advanced UI Project Files (.issuite)

Advanced UI and Suite/Advanced UI project files (.issuite) are XML-based files. The underlying XML schema for the user interface part of the file has changed significantly in InstallShield 2013. Many attributes have been moved to child elements. Conditions, actions, and validations are fundamentally different: element collections are used instead of attribute strings. The new schema is more explicit about what is configured for the user interface of the installation.

If you create a new Advanced UI or Suite/Advanced UI project in InstallShield 2013, InstallShield automatically uses the new XML schema for the .issuite file. If you upgrade a project from InstallShield 2012 Spring or earlier to InstallShield 2013, InstallShield automatically updates the XML schema.

New Predefined Path Variables for Built-in DLL Custom Actions

InstallShield includes two new predefined path variables in projects: ISRedistPlatformDependentFolder and ISRedistPlatformDependentExpressFolder. The default values for these folders refer to subfolders in the *InstallShield Program Files Folder*\Redist folder, which contains 32-bit versions of built-in InstallShield custom action DLLs. If you create a new InstallShield 2013 project or upgrade an InstallShield 2012 Spring or later project to InstallShield 2013, InstallShield automatically includes these two predefined path variables in your project. In addition, InstallShield also uses these path variables in paths such as the source location of built-in InstallShield DLL custom actions; this applies to new projects as well as upgraded projects. Previously, InstallShield used either

of the following locations as the folder that contained the DLL files: <ISProductFolder>\Redist\Language Independent\i386 or <ISProductFolder>\Redist\Language Independent\i386 Express. This change is noted for informational purposes.

Automation Interface Changes

If you use the automation interface with InstallShield or the Standalone Build, update your existing code to reflect the new ProgID: IswiAuto20.ISWiProject. The Standalone Automation Interface uses the same ISWiAutomation20.dll file that InstallShield uses, but it is installed to a different location.

Note that if you install the Standalone Build on the same machine as InstallShield, the last ISWiAutomation20.dll file that is registered is the one that is used.

Trialware Support

The only edition of InstallShield that includes the Trialware view is the Premier edition. This edition lets you create the Try and Die type of trialware. InstallShield no longer includes support for creating the Try and Buy/Product Activation type of trialware.

If you have an existing InstallShield Activation Service account and you want to be able to create the Try and Buy/Product Activation type of trialware in InstallShield 2013, you can still do so. For instructions, see Knowledge Base article Q200884.

Changes to the Default Wizard Format for the Suite/Advanced UI and Advanced UI Wizard Interface

If you create a new Suite/Advanced UI or Advanced UI project in InstallShield 2013, the default value for the Wizard Format setting is Glass, a new option. If you upgrade a Suite/Advanced UI or Advanced UI project from an earlier version of InstallShield to InstallShield 2013, InstallShield does not change the value of the Wizard Format setting; it leaves whatever value was selected in the earlier version of InstallShield.

You can configure your project to use to a different wizard format if appropriate. To do so, change the value of the Wizard Format setting that is displayed in the Wizard Interface view when the Wizard Pages node is selected.

Changes to the Font Color of Navigation Button Text in Advanced UI and Suite/Advanced UI Installations

In some scenarios, if you upgrade an Advanced UI or Suite/Advanced UI project from InstallShield 2012 Spring or earlier to InstallShield 2013, the text on the navigation buttons may no longer be legible: the colors for the text and the buttons may not have enough contrast. This may occur if a light font color is specified for the text style that is used for navigation button text, and if the target system's colors are configured to use a light color for buttons on windows.

Beginning with InstallShield 2013, Advanced UI and Suite/Advanced UI installations no longer ignore the font color that is specified for the text style of text that is used on navigation buttons. Thus, when you upgrade your Advanced UI or Suite/Advanced UI project to InstallShield 2013, you may need to adjust the color of the text style that is used for the navigation button text. To do so, in the Wizard Interface view, click the Wizard Pages node, and note the text style that is selected for the Navigation Text Style setting. Then, find that text style under the Styles node in this view, and adjust its settings as needed.

In InstallShield 2012 Spring or earlier, Advanced UI and Suite/Advanced UI projects ignored the text style color for navigation button text. At run time, the installation used the font color that Windows used for navigation buttons; the color was typically black.

New Support for Shell Properties of Shortcuts

The Shortcuts view in InstallShield now has built-in support for setting several Windows Shell properties of shortcuts:

- The Pin to Windows 8 Start Screen setting sets the System.AppUserModel.StartPinOption property by using the following GUID and property ID combination:
  9F4C2855-9F79-4B39-A8D0-E1D42DE1D5F3, 12
- The Prevent Pinning option in the Shell Properties setting sets the System.AppUserModel.PreventPinning property by using the following GUID and property ID combination:
  9F4C2855-9F79-4B39-A8D0-E1D42DE1D5F3, 9
- The Do Not Highlight as New option in the Shell Properties setting sets the System.AppUserModel.ExcludeFromShowInNewInstall property by using the following GUID and property ID combination:
  9F4C2855-9F79-4B39-A8D0-E1D42DE1D5F3, 8

Windows Installer may generate errors if one or more of these property names (instead of the GUID and property ID) are used in a package that is run on a version of Windows that does not have support for these properties.

If you used the Shell Properties setting in InstallShield 2012 Spring or earlier to configure Shell properties for a shortcut, and you upgrade the project to InstallShield 2013, InstallShield does not automatically replace the property names with the appropriate GUID and property ID combinations. To quickly switch to the GUID and property ID combination, consider deleting the old configuration in the Shell Properties view after upgrading your project, and then using the new support to reconfigure one or more of these properties. As an alternative, you can manually override the entry with the appropriate GUID and property ID.

This support is available for the following project types: Basic MSI, DIM, InstallScript MSI, Merge Module, MSI Database, MSM Database, and Transform.

Resolved Issues in InstallShield 2013 Original Release Version (June 2013)

IOA-000052899

If you launch InstallShield during the 21-day evaluation period while your machine is connected to a license server but a concurrent license is not free for you to use, InstallShield now displays an error stating that no licenses are currently available. Previously, InstallShield displayed an erroneous error stating that the machine could not connect to the license server.

IOA-000057280 (Basic MSI, DIM, InstallScript MSI, Merge Module)

The ProgId entries in the File Types subview for a component are now visible if more than 31 file extensions are displayed.

IOA-000061368

The IswiAuto version number in the sample automation interface script that is installed to *InstallShield Program Files Folder*\Samples\WindowsInstaller\Automation Interface Files\Add Files and Components has been updated for InstallShield 2013.

IOA-000061457

The value of the project reference name property in an .isproj file is no longer truncated if the Visual Studio solution name has more than 26 characters.

IOA-000062068 (Basic MSI, InstallScript MSI)

Using a Localized_Resource output group with a primary project output group no longer causes the .NET dependencies to be installed to the wrong destination; they are now installed to the same destination as the .NET assembly instead of to an sv subfolder. Previously, the dependencies were installed to the sv subfolder if the Dependencies and Properties option was selected for the .NET Scan at Build setting of a .NET assembly component.

IOA-000062851, IOA-000076726 (InstallScript)

When the ReleasePackager.exe command-line tool is creating a self-extracting Setup.exe launcher of an uncompressed InstallScript installation, it now copies the manifest from the uncompressed InstallScript installation and uses it for the self-extracting Setup.exe file.

As a result, the manifest that it embeds in the self-extracting Setup.exe launcher specifies the appropriate required execution level. Previously, the manifest always indicated Highest Available for the required execution level, regardless of what level was specified on the Setup.exe tab of the release in the Releases view.

In addition, the manifest also contains a compatibility section. Previously, without this compatibility section, a Program Compatibility Assistant (PCA) dialog box may have appeared at the end of an installation on some target systems. The PCA dialog box indicated that the program might not have installed correctly.

IOA-000065719

If you press F1 when the insertion point is in an InstallScript function or other keyword in the InstallScript view, it opens the help topic for that function or keyword. Previously, an unexpected help topic was displayed for some keywords.

IOA-000068598

A note for InstallScript MSI installations was added to the "Event Handlers" help topic. The note indicates that when an InstallScript MSI major upgrade is uninstalling an earlier version of a product, none of the InstallScript event handlers are called.

IOA-000069628 (Basic MSI, InstallScript MSI)

The Setupini.exe file, a command-line tool that lets you modify the setup.ini file that is embedded in a Setup.exe or Update.exe file, has been updated for InstallShield 2013. The InstallShield installation installs Setupini.exe to the *InstallShield Program Files Folder*\System folder. Previously, this tool was available only from a Knowledge Base article, and the InstallShield 2012 version was not able to open compressed Setup.exe files that were built in InstallShield 2012.

IOA-000069928 (Basic MSI, InstallScript MSI)

Inserting a SQL script file that has a long file name into the SQL Scripts view no longer causes InstallShield to stop responding.

IOA-000071278

If a Visual Studio solution contains an InstallShield project that was upgraded from an earlier version of InstallShield, and if builds are done through Team Foundation Server (TFS), build error -5056 no longer occurs. In addition, MSBuild no longer fails with the error, "The program can't start because BECommonLib.dll is missing from your computer." Previously if you upgraded InstallShield projects that were created in an earlier version of InstallShield to a later version, these issues occurred because the .isproj file was not updated correctly.

IOA-000071280

If a Visual Studio solution contains an InstallShield project that was upgraded from an earlier version of InstallShield, and if builds are done through Team Foundation Server (TFS), the build now uses the MSBuild targets

file for the current version of InstallShield. Previously, the build failed because it was referencing the MSBuild targets file for the earlier version of InstallShield.

IOA-000071481

The correct umlaut characters are now used in the Hungarian run-time string for the Hungarian version of "Another instance of this setup is already running. Please wait for the other instance to finish and then try again."

IOA-000072567 (Basic MSI, InstallScript, InstallScript MSI, Merge Module, Suite/Advanced UI)

If you define a command for the Prebuild Event setting for a release in the Releases view and then click a different release in the project, the command that you entered for the first release's Prebuild Event setting is no longer automatically entered for the Prebuild Event setting of the second release.

IOA-000072615 (Advanced UI, Suite/Advanced UI)

If you build a release for a Suite/Advanced UI project or an Advanced UI project, and the project contains a feature that does not contain any packages, build warning -7337 ("Feature [1] has no associated packages.") is displayed. To resolve this build warning, ensure that at least one package is associated with the feature. If you intentionally are not including a package with the feature that is identified in the build warning, you can ignore this build warning.

IOA-000072662 (Advanced UI, Suite/Advanced UI)

If a subfeature is configured to be hidden on the feature selection wizard page and an end user deselects its parent feature, the installation no longer installs the subfeature.

IOA-000072747 (Advanced UI, Suite/Advanced UI)

If you add a progress circle control to a wizard page in an Advanced UI or Suite/Advanced UI installation, it is now displayed at run time.

IOA-000072888 (Advanced UI, Suite/Advanced UI)

If you specify a local host location in the Update URL setting for an Advanced UI or Suite/Advanced UI project, put the base installation in that location, and launch a copy of the base installation from the target system, the installation no longer keeps calling itself and getting caught in an infinite loop.

IOA-000072902 (Basic MSI, DIM)

If you use a release flag to exclude a feature that has a shortcut that configures Windows Shell properties, the release flag now excludes the Windows Shell property data from builds. Previously, entries for the excluded shortcut were included in the MsiShortcutProperty table.

IOA-000072948 (Advanced UI, Suite/Advanced UI)

If you run an Advanced UI or Suite/Advanced UI installation that runs an uncompressed package, the fields for the create and modified dates of the files that the package installs are no longer changed to the time of the installation.

IOA-000073033 (Advanced UI, Basic MSI, InstallScript, InstallScript MSI, Suite/Advanced UI)

The conditions of the following InstallShield prerequisites have been changed to allow them to be run on 64-bit target systems:

- Microsoft Visual C++ 2008 SP1 Redistributable MFC Security Update KB2538243 (x86)
- Microsoft Visual C++ 2005 SP1 Redistributable MFC Security Update KB2538242 (x86)

- Microsoft Visual C++ 2005 SP1 Redistributable Package (x86)

Previously, the conditions prevented these InstallShield prerequisites from being run on 64-bit systems.

IOA-000073047 (InstallScript, InstallScript MSI)

The Browse button on the German version of the SQLServerLogin dialog in an installation that uses the Blue or BlueTC dialog skin is no longer distorted. Previously the Browse button was positioned on a radio button control.

IOA-000073056 (Basic MSI, InstallScript MSI)

If you resize the Run-time Message column in the list of required software on the Installation Requirements page in the Project Assistant, InstallShield no longer stops responding.

IOA-000073186 (Basic MSI, InstallScript MSI)

The following can now be extracted from a compressed Setup.exe file for a Basic MSI or InstallScript MSI project and launched: (a) a compressed .msi package that contains files whose total combined size is large but not more than 2 GB and (b) an InstallShield prerequisite that is compressed into the Setup.exe file. Previously, these files could not be extracted from Setup.exe, and run-time error 1152 occurred.

Note that an .msi package cannot be more than 2 GB.

IOA-000073984

The "Calling a PowerShell Custom Action" and "Windows Installer Property Reference" help topics in the InstallShield help library now reference the POWERSHELLVERSION property. Previously, they referred to an ISPOWERSHELLINSTALLED property instead of POWERSHELLVERSION.

IOA-000074184 (InstallScript, InstallScript MSI)

If you add a list box control to a dialog, the Item setting is no longer one of the settings that is displayed when you select that control. The Item setting is not applicable for this control in InstallScript or InstallScript MSI projects.

IOA-000074187 (Basic MSI, InstallScript MSI)

Including more than one PowerShell custom action in an installation no longer causes the installation to fail at run time.

IOA-000074468 (Basic MSI, InstallScript MSI, QuickPatch)

It is not possible to create an uninstallable patch if the patch adds new records to the Windows Installer tables that configure InstallShield's support for COM+, IIS, XML file changes, text file changes, network shares, or SQL script changes. Therefore, VAL0015 now warns if you perform validation for an upgrade that adds new records to these tables.

IOA-000074603 (Advanced UI, Basic MSI, InstallScript, InstallScript MSI, Suite/Advanced UI)

The conditions and the product GUID in the .prq file for the Microsoft Office 2010 Primary Interop Assemblies (PIA) redistributable have been updated to reflect the current redistributable that Microsoft has available on their Web site. Previously, the conditions and the product GUID were wrong, leading to unexpected behavior at run time.

IOA-000074624 (Basic MSI, InstallScript MSI)

If you perform ICE24 validation for an .msi package that contains an invalid product code, an ICE24 error is reported.

IOA-000074812, IOA-000074888 (Advanced UI, Basic MSI, InstallScript, InstallScript MSI, Suite/Advanced UI)

Neither InstallShield prerequisite that installs the .NET Framework 4.5 no longer fails if the installation that includes the prerequisite is configured to download the .NET Framework redistributable from the Web.

IOA-000074889 (Advanced UI, Suite/Advanced UI)

After a target system has been restarted for a package in an Advanced UI or Suite/Advanced UI installation that was launched silently, the Advanced UI or Suite/Advanced UI installation resumes silently. (Note that if any remaining part of the installation requires elevated privileges, a UAC prompt is displayed.)

Previously in that scenario, after the restart, the user interface of the Advanced UI or Suite/Advanced UI installation was displayed; that is, the installation did not resume silently.

IOA-000075032 (Basic MSI, InstallScript, InstallScript MSI)

If a call to the LaunchApplication function is successful and the LAAW_OPTION_WAIT option was specified, the LAAW_PARAMETERS system variable's nLaunchResult member contains the return code of the launched application. Previously, LAAW_PARAMETERS.nLaunchResult always returned the number 0.

IOA-000075214 (Advanced UI, Basic MSI, InstallScript, InstallScript MSI, Suite/Advanced UI)

If an installation includes the InstallShield prerequisite that installs the Microsoft VSTO 2010 Runtime and the location of the prerequisite is configured to be downloaded from the Web, the installation no longer fails with the message, "The files for installation requirement Microsoft VSTO 2010 Runtime could not be found." The prerequisite's .prq file now references the name of the VSTO runtime installation that is currently posted on Microsoft's site. Previously, the prerequisite's .prq file referenced the old name of the file on Microsoft's site.

IOA-000075403, IOA-000076787

If you are using InstallShield from within Visual Studio 2012, InstallShield now uses 4.0 for the value of the ToolsValue attribute in the .isproj file. If you use MSBuild or Team Foundation Server (TFS) to build your release, the build no longer fails with error MSB4062 reporting that the Microsoft.Build.Tasks.AssignProjectConfiguration task could not be loaded from the assembly Microsoft.Build.Tasks. In addition, the build no longer fails with error MSB3202 reporting that the project file was not found. Previously, the .isproj file contained the wrong value for the ToolsValue attribute, causing these types of errors.

IOA-000075404

If you are using InstallShield from within Visual Studio 2012 for a C++ project, InstallShield now correctly sets the FileName field of the File table record for the primary output of the C++ project. If you use MSBuild or Team Foundation Server (TFS) to build your release, the build no longer fails with build error -5023 (Error building table File).

IOA-000075905 (Basic MSI, InstallScript MSI, QuickPatch)

If you try to build a patch that is configured to automatically generate entries for the MsiPatchOldAssemblyFile and MsiPatchOldAssemblyName tables, and if the patch should contain an assembly file that needs to be patched and whose manifest is not set, the build no longer fails with fatal error -7071.

IOA-000076005 (Advanced UI, Suite/Advanced UI)

If the list of features that are displayed on the InstallationFeature wizard page requires the use of a scroll bar, the top of the list of features is now initially displayed. Previously, the bottom of the list was initially displayed.

IOA-000076255 (Advanced UI, Suite Advanced UI)

A compressed Setup.exe file for an Advanced UI or Suite/Advanced UI project can now extract packages whose total combined size is larger than 2 GB. Previously, the packages could not be extracted from Setup.exe, and a run-time error about extracting setup.xml occurred.

IOA-000076485

If you use MSBuild or TFS to build a release for an InstallShield project that is part of a Visual Studio solution, the resulting installation now includes the same dependencies as an installation that was generated from within Visual Studio.

IOA-000076538

If you install InstallShield on an x64 machine, the InstallShield installation now sets the default paths for Regasm.exe and InstallUtilLib.dll. Previously, the paths were not set by default, and the path had to be specified manually on the .NET tab of the Options dialog box (available from the Tools menu) or through the registry directly (DotNetInstallUtilLibPath64 and DotNetRegasmPath64 values under HKLM\SOFTWARE\Wow6432Node\InstallShield*Version*\Professional).

System Requirements

This section contains the minimum requirements for systems that run InstallShield (the authoring environment), as well as for target systems that run the installations created with InstallShield (the run-time environment).

For Systems Running InstallShield

Processor

Pentium III-class PC (500 MHz or higher recommended)

RAM

256 MB of RAM (512 MB preferred)

Hard Disk

500 MB free space

Display

Designed for XGA resolution at 1024 × 768 or higher

Operating System

Windows Vista
Windows Server 2008
Windows 7
Windows Server 2008 R2
Windows 8
Windows Server 2012
Windows 8.1
Windows Server 2012 R2

Privileges

Administrative privileges on the system

Mouse

Microsoft IntelliMouse or other compatible pointing device

Optional Integration with Visual Studio

The following versions of Microsoft Visual Studio can be integrated with InstallShield 2013 Premier or Professional Editions:

Visual Studio 2008
Visual Studio 2010
Visual Studio 2012
Visual Studio 2013

The following editions of these versions of Visual Studio can be integrated with InstallShield 2013 Premier or Professional Editions:

Professional
Premium
Ultimate

For Target Systems

Target systems must meet the following minimum operating system requirement:
Windows XP (Windows XP SP3 for Advanced UI and Suite/Advanced UI installations)
Windows Server 2003 (Windows Server 2003 SP2 for Advanced UI and Suite/Advanced UI installations)
Windows Vista
Windows Server 2008
Windows 7
Windows Server 2008 R2
Windows 8
Windows Server 2012
Windows 8.1
Windows Server 2012 R2

Known Issues

For a list of known issues, see Knowledge Base article Q210473.