InstallShield 2012 Spring

Release Notes

originally released May 2012; updated to include SP1, released September 2012

Introduction

InstallShield is the industry standard for authoring high-quality Windows Installer– and InstallScript-based installations, as well as Microsoft App-V packages. InstallShield 2012 Spring offers new features and enhancements that make it easy to use the latest technologies and create a modern look and feel for your installation's user interface.

InstallShield 2012 Spring includes support for the latest technologies, such as Windows 8, Windows Server 2012, Visual Studio 2012, .NET Framework 4.5, and SQL Server 2012. It also lets you create installations that run SQL scripts on SQL Azure database servers.

The new Advanced UI installation support in the Professional edition of InstallShield 2012 Spring enables you to create and customize a new end-user interface, with redesigned, contemporary wizard pages, for a Windows Installer package or an InstallScript package. This new project type is based on the same technology that was previously introduced as the Suite project type (now known as the Suite/Advanced UI project type) in the Premier edition of InstallShield.

InstallShield 2012 Spring Service Pack 1 (SP1) includes changes that offer support for the final released versions of Windows 8, Windows Server 2012, and Visual Studio 2012. (Note: Without SP1, InstallShield 2012 Spring supports Windows 8 Consumer Preview, Windows Server 2012 Beta, and the beta of Visual Studio 2012.)

For the latest information about InstallShield 2012 Spring, including updates to these release notes, see Knowledge Base article Q204468.

Changes in SP1 (September 2012)

To obtain SP1, see KB article Q208466.

Upgrade Note: If you want to open an InstallShield 2012 Spring Advanced UI or Suite/Advanced UI project (.issuite) in InstallShield 2012 Spring SP1, you must allow InstallShield to upgrade your project to InstallShield 2012 Spring SP1. InstallShield 2012 Spring SP1 Advanced UI and Suite/Advanced UI projects include support for functionality that was not available in these project types in InstallShield 2012 Spring SP1, and this support needs to be added during the upgrade. Note that it is not possible to open InstallShield 2012 Spring SP1 Advanced UI or Suite/Advanced UI projects in earlier versions of InstallShield (including InstallShield 2012 Spring without SP1). Therefore, if multiple users need to open and modify your InstallShield projects, ensure that all of you apply the SP1 patch at the same time.

If you open an InstallShield 2012 Spring Advanced UI or Suite/Advanced UI project in InstallShield 2012 Spring SP1, InstallShield 2012 Spring SP1 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 before converting it.

Support for Adding Sideloading App Packages (.appx) to Suite/Advanced UI Projects

InstallShield now lets you add sideloading app packages (.appx) to a Suite/Advanced UI project. You can add this package type through the Packages view of the Suite/Advanced UI project just as you would add other types of packages to the project.

Sideloading an app is the process of installing an app without obtaining it through the Windows Store. This type of app is sometimes distributed to enterprise environments. Windows 8 and Windows Server 2012 include support for sideloading apps.

This support is available in the Premier edition of InstallShield.

Note that support for creating and building Suite/Advanced UI installations that include sideloading app packages (.appx) requires Windows Vista or later or Windows Server 2008 or later on the machine that has InstallShield or the Standalone Build.

New AppX Package Type of Condition Check in 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, you can select from a number of different types of checks that you want to be evaluated on target systems. Use the new AppX Package type of condition check to check target systems for the presence of a particular .appx package. The condition checks for a particular app name, and it can also check other information, such as the version.

Support for Visual Studio 2012, .NET Framework 4.5, and Visual C++ 2012

InstallShield includes changes that offer support for the final released version of Visual Studio 2012, enabling development of installations and products within this version of the Visual Studio interface.

In addition, InstallShield includes two updated InstallShield prerequisites for the .NET Framework and two new InstallShield prerequisites for Visual C++:

- Microsoft .NET Framework 4.5 Full
- Microsoft .NET Framework 4.5 Web
- Microsoft Visual C++ 2012 Redistributable Package (x86)
- Microsoft Visual C++ 2012 Redistributable Package (x64)

You can add any of these prerequisites to Advanced UI, Basic MSI, InstallScript, InstallScript MSI, and Suite/Advanced UI projects.

The Web prerequisite for the .NET Framework requires an Internet connection. This prerequisite downloads the required redistributable files if appropriate. The full prerequisite is a stand-alone installation that does not require an Internet connection.

IOA-000062896

If you upgrade an InstallShield project that was created with an earlier version of InstallShield to the latest version, if the project includes an MSBuild .isproj file, and if the Project element does not contain a ToolsVersion attribute, InstallShield now adds the ToolsVersion attribute.

IOA-000063001

The "AddProperty Method" in the InstallShield Help Library has been expanded. It now includes sample code that demonstrates how to use the AddProperty method with the automation interface. It also shows the method's syntax and the parameter that it uses.

IOA-000066198 (Advanced UI, Suite/Advanced UI)

If you edit a string entry in a project, the Modified date column shows the date and time when the change was made. If you save the project, close it, and reopen it, the modified date and time are now retained. Previously, an incorrect date and time were displayed.

IOA-000066199, IOA-000070351 (Basic MSI, DIM, InstallScript MSI, and Merge Module)

The kernel mode monitoring method for COM extraction now works with proxy stub DLLs and other files that attempt to perform per-user registration.

IOA-000068616

In the InstallShield Help Library, the sample InstallScript code for the CtrlGetSubCommand function has been corrected.

IOA-000069685

The sample code in the "ISWiUpgradeTableEntries Collection" help topic in the InstallShield Help Library has been corrected.

IOA-000070276 (Advanced UI, Suite/Advanced UI)

A new help topic called "Referring to Feature States and Other Feature Data in the UI of an Advanced UI or Suite/Advanced UI Project" is available in the InstallShield Help Library. This help topic explains how to trigger specific UI behavior that depends on feature-related information.

IOA-000070629 (Basic MSI)

If you configure your project to include a software identification tag with your product, and if you specify a .pfx file for the digital signature information for your release, InstallShield no longer generates build error -1027, indicating that the .swidtag could not be signed.

IOA-000070679 (Basic MSI, InstallScript MSI)

If you add the PowerShell system search to your project through the Installation Requirements page of the Project Assistant, there is no longer a display issue for that setting. Previously, if you closed and then reopened the project, the PowerShell system search was no longer selected in the Project Assistant.

IOA-000070856, IOA-000073213 (Basic MSI)

If more than one feature in your project contains a unique DIM reference, InstallShield now associates each DIM reference with the appropriate feature in the installation that it builds at build time. Previously, InstallShield built the release incorrectly in this scenario, and DIM references were associated with the wrong features.

IOA-000071301 (InstallScript)

When an end user uses the /hide_progress comamnd-line parameter to launch an InstallScript installation from the command line, the SdWelcome dialog is now displayed in front of the Command Prompt window instead of behind it.

IOA-000072344 (InstallScript)

The help topic "Running an InstallScript Installation Multiple Times" has been corrected. It now states that when you run a multi-instance installation silently on a machine that already has the product installed, the installation suppresses the Qualifying Product(s) Detected dialog and creates a new instance on the machine.

IOA-000072753 (Advanced UI, Suite/Advanced UI)

The checked list box control for wizard pages has been revised. To specify the labels for the check boxes that you want to be displayed in this control, and to associate the enabled state of each check box with the value of a property, use syntax such as the following in the Property setting for the checked list box control:

ID_Check1\r*PropertyName1*\n*ID_Check2*\r{Binding *PropertyName2*==*PropertyValue2*}\n*ID_Check3*\r{Binding *PropertyName3*!=*PropertyValue3*}

ID_Check1 corresponds with *PropertyName1*; *ID_Check2* corresponds with *PropertyName2* and *PropertyValue2*; and *ID_Check3* corresponds with *PropertyName3* and *PropertyValue3*.

\r is the separator between the name of each check box label and its binding.

\n is the separator between label-binding sets.

In this example, the *ID_Check1* check box is selected if the value of *PropertyName1* is not empty; selecting this check box sets *PropertyName1* to true (and clearing the check box clears the property value).

The *ID_Check2* check box is selected if the value of *PropertyName2* is equal to *PropertyValue2*; selecting this check box sets the value of *PropertyName2* to *PropertyValue2* (and clearing the check box clears the property value).

The *ID_Check3* check box is selected if the value of *PropertyName3* is not equal to *PropertyValue3*; clearing this check box sets the value of *PropertyName3* to *PropertyValue3* (and selecting it clears the property value).

IOA-000072845 (Basic MSI, InstallScript MSI)

If your installation creates an IIS application pool identity that was configured in the Internet Information view, the installation no longer includes the application pool's identity password in the Windows Installer log file when the password is stored as the value of a Windows Installer property.

IOA-000073285 (Advanced UI, Suite/Advanced UI)

If you use the IsCmdBld.exe command-line parameter -y to override the product version when building a release, IsCmdBld.exe now uses the version number that you specify for the properties of the setup launcher file. Previously, IsCmdBld.exe used the product version that was specified in the project file (.issuite).

IOA-000073286 (Basic MSI, InstallScript MSI)

If two projects use the same splash screen and you build compressed Setup.exe releases from those projects simultaneously, build error -6003 (An error occurred streaming '%1' into setup.exe.) no longer occurs. Previously, this build error occurred occasionally because of a sharing violation.

IOB-000061850 (Basic MSI, DIM, InstallScript MSI, Merge Module)

If you use the kernel mode monitoring method for COM extraction for a 32-bit file on a 64-bit system, the resulting registry entries no longer contain Wow6432Node.

IOC-000057212

The "Adding the Ability to Create or Set an Existing User Account" help topic now contains information about the limitations of the LogonInformation dialogs. This help topic explains that in some scenarios, when end users attempt to use the LogonInformation dialogs, they may encounter a blank list of domains, or they may encounter a

"Server not found" error. The help topic lists scenarios that may cause this behavior. For example, this issue may occur if the target system is not on a domain.

IOC-000063169

A new "Using Build Status Events" help topic is available in the InstallShield Help Library. This help topic contains sample code that demonstrates how to use the build status events ProgressIncrement, ProgressMax, and StatusMessage to show the progress of the build and see status updates when you are building a release through the automation interface.

IOC-000069203

The "ISWiPathVariables Collection" help topic contains new sample code that demonstrates how to use the ISWiPathVariables collection through the automation interface to retrieve a list of every path variable in the project.

IOC-000088682

The broken links in DIFx help topics for InstallScript functions such as DIFxDriverPackageGetPath, DIFxDriverPackageInstall, and DIFxDriverPackagePreinstall has been fixed. Previously, these help topics contained a broken link for "DIFx Errors."

New Features in InstallShield 2012 Spring Original Release Version (May 2012)

Ability to Target Windows 8 and Windows Server 2012 Systems

InstallShield enables you to specify that your installation requires Windows 8 or Windows Server 2012. It also lets you build feature and component conditions for these operating systems.

The InstallShield prerequisites that should be installable on Windows 8 and Windows Server 2012 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
- Microsoft ReportViewer 2010
- Microsoft SQL CE 3.5 SP2
- Microsoft SQL Server 2005 Express SP3
- Microsoft SQL Server 2008 Express SP1
- Microsoft SQL Server 2008 Management Objects 10.00.2531
- Microsoft SQL Server 2008 Native Client 10.00.2531
- Microsoft SQL Server 2008 R2 Express RTM
- Microsoft SQL Server 2008 R2 Native Client 10.50.1600.1
- Microsoft SQL Server Native Client 9.00.4035
- Microsoft SQL Server System CLR Types 10.00.2531
- Microsoft Visual C++ 2005 SP1 Redistributable MFC Security Update KB2538242
- Microsoft Visual C++ 2005 SP1 Redistributable Package
- Microsoft Visual C++ 2008 SP1 Redistributable MFC Security Update KB2538243
- Microsoft Visual C++ 2008 SP1 Redistributable Package
- Microsoft Visual C++ 2010 Redistributable Package

- Microsoft Visual C++ 2010 RTM Redistributable MFC Security Update KB2467173
- Microsoft Visual C++ 2010 SP1 Redistributable Package
- Microsoft VSTO 2010 Runtime

Support for a New Contemporary, Customizable End-User Interface in InstallShield Professional Edition

The new Advanced UI project type lets you create a new end-user interface, with redesigned, contemporary wizard pages, for a Windows Installer package or an InstallScript package. This new project type is available in the Professional edition of InstallShield, and it is based on the technology that was previously introduced as the Suite project type (now known as the Suite/Advanced UI project type) in the Premier edition of InstallShield.

The new Advanced UI project type includes built-in wizard pages that you can include and customize in your Advanced UI installations. The wizard page editor in this project type lets you add, sequence, and remove pages as needed; it also lets you modify the layout of any page—adding, moving, and removing a variety of different kinds of controls. All of the new UI functionality that was previously available only through the Premier edition is now available through Advanced UI projects.

Like Suite/Advanced UI projects, an Advanced UI project uses the next-generation setup launcher (Setup.exe) for launching a package on target systems. Also like Suite/Advanced UI projects, an Advanced UI project includes flexible options for specifying the run-time source location of the package that you are including in the Advanced UI installation. The available location options are:

- On the Web, available for download by Setup.exe if needed
- Embedded in Setup.exe and extracted to the target system if needed
- Uncompressed and stored on the Advanced UI source media

Your end users can quickly download a small Advanced UI Setup.exe file, and the Setup.exe file can download and launch the Windows Installer–based or InstallScript package if needed.

Note that the Advanced UI project type includes support for including only one primary .msi package, one primary .msp package, or one primary InstallScript package. The Suite/Advanced UI project type that is available in the Premier edition of InstallShield includes support for packaging multiple primary installations (including .exe packages) as a single installation.

Microsoft SQL Azure Support

InstallShield now includes support for running SQL scripts on Microsoft SQL Azure database servers. In addition, InstallShield includes Microsoft SQL Azure in the predefined list of database servers that you can select when you are specifying in the SQL Scripts view the target database servers that your product supports.

If your installation targets SQL Azure, the SQLBrowse run-time dialog that is displayed when end users choose to browse for a database catalog can now list catalogs on the specified SQL Azure database server.

This support is available in the following project types: Basic MSI, DIM, InstallScript, and InstallScript MSI.

Beta Support for Microsoft Visual Studio 2012

InstallShield includes support for the beta of Visual Studio 2012. You can create InstallShield projects from within this version of Visual Studio.

Microsoft .NET Framework 4.5 Prerequisites

InstallShield includes two new .NET-related InstallShield prerequisites that you can add to Basic MSI, InstallScript, and InstallScript MSI projects:

- Microsoft .NET Framework 4.5 Full
- Microsoft .NET Framework 4.5 Web

These InstallShield prerequisites install the beta versions of the .NET Framework 4.5 on supported target systems. The Web prerequisite requires an Internet connection. This prerequisite downloads the required redistributable files if appropriate. The full prerequisite is a stand-alone installation that does not require an Internet connection.

Microsoft SQL Server 2012 Support

InstallShield now includes support for running SQL scripts on SQL Server 2012 database servers. In addition, InstallShield includes SQL Server 2012 in the predefined list of database servers that you can select when you are specifying in the SQL Scripts view the target database servers that your product supports.

If your installation targets SQL Server 2012, the SQLBrowse run-time dialog that is displayed when end users choose to browse for a database server can now list instances of SQL Server 2012, SQL Server 2012 Express, and SQL Server 2012 Express LocalDB. In addition, the SQLBrowse run-time dialog that is displayed when end users choose to browse for a database catalog can now list catalogs on the specified SQL Server 2012 database server.

This support is available in the following project types: Basic MSI, DIM, InstallScript, and InstallScript MSI.

Microsoft SQL Server 2012 Prerequisites

InstallShield includes several new SQL Server 2012–related InstallShield prerequisites that you can add to Basic MSI, InstallScript, and InstallScript MSI projects:

- Microsoft SQL Server 2012 Express
- Microsoft SQL Server 2012 Express LocalDB
- Microsoft SQL Server 2012 Native Client

InstallShield also includes InstallShield prerequisites that install Microsoft .NET Framework 3.5 SP1 Update KB956250, which is a dependency of Microsoft SQL Server 2012 Express.

These InstallShield prerequisites install the technology on supported target systems.

New InstallShield Prerequisites for App-V 4.6 SP1, SQL Server Compact 4.0, and JRE SE 1.7

InstallShield includes new InstallShield prerequisites that you can add to Basic MSI, InstallScript, and InstallScript MSI projects:

- Java Runtime Environment Second Edition (JRE SE) 1.7
- Microsoft App-V 4.6 SP1 Desktop Client (The redistributable files for these InstallShield prerequisites—available in 32-bit and 64-bit versions—are not available for download from within InstallShield, since you must obtain them from Microsoft. Once you obtain them from Microsoft, place them in the location that is displayed when you are editing these prerequisites in the InstallShield Prerequisite Editor.)
- SQL Server Compact 4.0

These InstallShield prerequisites install the technology on supported target systems.

This feature resolves the following issues: IOA-000061260, IOA-000066507.

Support for the Configuring System Center 2012 Configuration Manager Application Model Data

Accurate identification of deployment metadata is necessary for migrating applications into the System Center 2012 Configuration Manager application model. InstallShield includes several new settings in the General Information view that let you specify some of the application model metadata for an application through a software identification tag. When AdminStudio users import a package into the AdminStudio Application Catalog,

AdminStudio Application Manager mines package elements for deployment data such as detection methods, dependencies, requirements, as well as information in the software identification tag. AdminStudio makes this information available to users for review and tests before publishing to Microsoft System Center 2012 Configuration Manager.

This feature is available in Basic MSI projects.

Support for PowerShell Custom Actions

Windows PowerShell is a .NET Framework–based command-line shell and script language that enables system administrators to automate system configuration tasks. InstallShield now has support for custom actions that run PowerShell scripts. You may want to add this type of custom action to a project to perform system configuration tasks at installation run time.

Note that in order for an installation to run a PowerShell custom action, Windows PowerShell must be installed on target systems. InstallShield includes a new predefined PowerShell system search that checks for the presence of PowerShell on target systems. You can include this system search in your project and configure your PowerShell custom action to run only if the system search determines that PowerShell is installed.

The PowerShell execution policy, which determines whether PowerShell scripts can be run on a target system, is set to restricted by default, which does not permit PowerShell scripts to be run. If you want your installation to override the target system's execution policy with an appropriate one for your installation's PowerShell custom actions, you can use the new Windows Installer property IS_PS_EXECUTIONPOLICY to indicate the appropriate execution policy.

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

This feature resolves issue IOA-000047578.

Automatic Update Check and Download Support for Suite/Advanced UI and Advanced UI Installations

Suite/Advanced UI and Advanced UI installations now have the ability to automatically check for an updated Suite/Advanced UI or Advanced UI Setup.exe file that you host on your Web site, and download and launch it if it is available. The updated Suite/Advanced UI or Advanced UI Setup.exe file can be used to deploy upgrades and patches for your latest Suite/Advanced UI and Advanced UI packages.

The Setup.exe tab in the Releases view of Suite/Advanced UI and Advanced UI projects has a new Update URL setting that lets you specify the absolute path (including the file name) for an updated Suite/Advanced UI or Advanced UI setup launcher. If the base Suite/Advanced UI or Advanced UI setup launcher is running a non-maintenance operation, the Suite/Advanced UI or Advanced UI setup launcher checks the update URL for a download. If a download is available, the base Suite/Advanced UI or Advanced UI setup launcher downloads it and then verifies its digital signature. If the digital signature in the update setup launcher matches that in the base setup launcher, the update setup launcher runs automatically. If the digital signature does not match, or if the base setup launcher is not digitally signed, a security warning is displayed, allowing the end user to choose whether to proceed with the update setup launcher.

Ability to Detect Whether a Specific Version of a Suite/Advanced UI or Advanced UI Installation Is Already Installed

Suite/Advanced UI and Advanced UI projects now include support for determining whether a particular version of a Suite/Advanced UI or Advanced UI installation is already installed on target systems. This type of condition check is called a Suite Installed condition.

InstallShield now includes two Suite Installed conditions in each Suite/Advanced UI and Advanced UI project by default:

- A new Suite Installed exit condition prevents end users from being able to install the current version of the Suite/Advanced UI or Advanced UI installation over a future newer version of the same Suite/Advanced UI or Advanced UI installation.
- A new Suite Installed mode condition may prevent the installation from running in first-time installation mode if the same version of the Suite/Advanced UI or Advanced UI installation is already installed.

These new default conditions are available in all new Suite/Advanced UI and Advanced UI projects. If you upgrade an InstallShield 2012 Suite project to InstallShield 2012 Spring, InstallShield automatically adds these default conditions to the project.

You can edit the default Suite Installed conditions if necessary. You can also add your own Suite Installed conditions to a project as needed.

Ability to Configure Network Sharing of Folders in an Installation

InstallShield now lets you create, modify, and delete network shares to enable file sharing over networks. When you are configuring a folder in your project, you can indicate that you want to enable network sharing, which is disabled by default. You can also configure other options such as the name of the share, as well as the maximum number of simultaneous users who can access the share.

To enable sharing, use the new Sharing tab on the Properties dialog box, which is displayed when you right-click a folder in the Files and Folders view and then click Properties.

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

New Built-In Custom Action that Terminates Specific Processes

InstallShield includes support for a new kill-process type of custom action. If you add this type of custom action to your project, you can specify the name or process identifier (PID) of one or more processes that you want to be terminated at run time, and you can schedule the custom action for immediate or deferred mode.

The procedure for creating this type of custom action involves adding and configuring the custom action in the Custom Actions and Sequences view of your project, and using the Property Manager view to define a property with the names or PIDs of the appropriate processes.

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

Ability to Create Predetermined User Accounts and Groups at Run Time

InstallShield now has built-in support for creating multiple user accounts and corresponding groups at run time without using logon dialogs. To configure the accounts and groups, define values for the properties ISNetApiLogonUsername, ISNetApiLogonGroup, and ISNetApiLogonPassword in your project with the user names, groups, and passwords, respectively. Separate multiple names, groups, or passwords with a tilde enclosed by square brackets: [~].

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

This feature resolves part of issue IOA-000054311.

Ability to Include InstallShield Prerequisites as Packages in Suite/Advanced UI and Advanced UI Projects

InstallShield now lets you import InstallShield prerequisites as .msi and .exe packages into Suite/Advanced UI and Advanced UI projects. You can import the InstallShield prerequisites that are included with InstallShield, as well as any custom InstallShield prerequisites that you have created. When you right-click the Packages explorer in the Packages view and then click the new Import Prerequisite (.prq) command, InstallShield adds to your project an .msi package or an .exe package, depending on the type of file that is configured to run for the prerequisite. InstallShield also automatically configures default values for each of the prerequisite package's settings, based on the settings that are configured in the .prq file. You can change these settings as needed, just as you can change the settings for packages in your Suite/Advanced UI or Advanced UI project.

If an InstallShield prerequisite has dependencies (that is, if one or more other .prq files are specified as dependencies in the InstallShield prerequisite that you are adding to the Suite/Advanced UI or Advanced UI project), InstallShield automatically adds the dependency prerequisites as separate packages in the Packages explorer.

Previously it was necessary to add the .msi and .exe installations as packages in a Suite project and then manually configure all of the conditions and settings for each of those packages.

This feature resolves issue IOA-000065350.

Ability to Include InstallScript Installations as Packages in Suite/Advanced UI and Advanced UI Projects; New Suite/Advanced UI– and Advanced UI–Specific InstallScript Events and Functions

InstallShield now lets you add InstallScript installations as packages in Suite/Advanced UI and Advanced UI projects. When a Suite/Advanced UI or Advanced UI installation launches an InstallScript package, the Suite/Advanced UI or Advanced UI installation displays its own user interface (UI) while automatically suppressing the UI of the InstallScript package. This enables you to provide a contemporary UI experience for the installation. The Suite/Advanced UI or Advanced UI installation also displays progress information for the InstallScript package.

To make these changes possible, Suite/Advanced UI and Advanced UI installations use several new Suite/Advanced UI– and Advanced UI–specific InstallScript events and functions by default, and ignores some of the standard InstallScript events and functions.

New InstallScript Package Support in Suite/Advanced UI and Advanced UI Projects

InstallShield lets you add an InstallScript package to a Suite/Advanced UI or Advanced UI project if the InstallScript package meets the following requirements:

- The InstallScript package is uncompressed.
- InstallShield 2012 Spring or later is used to build the InstallScript package and the Suite/Advanced UI or Advanced UI installation.
- The InstallScript package uses an event-based script; it should not use the program...endprogram style script.

New Suite/Advanced UI– and Advanced UI–Specific InstallScript Events and Functions

In a standard InstallScript installation that is launched via the InstallScript Setup.exe file (that is, not launched from a Suite/Advanced UI or Advanced UI installation), most events are called directly from the OnShowUI event. In a Suite/Advanced UI or Advanced UI installation that launches an InstallScript package, OnShowUI is replaced with OnSuiteShowUI. Depending on the installation state (first-time installation, maintenance, or update), OnSuiteShowUI ignores the UI events such as OnFirstUIBefore and OnFirstUIAfter and instead calls the following events:

- First-time installation—OnSuiteInstallBefore, OnSuiteInstallAfter
- Maintenance—OnSuiteMaintBefore, OnSuiteMaintAfter
- Update—OnSuiteUpdateBefore, OnSuiteUpdateAfter

The InstallScript language includes some new Suite/Advanced UI– and Advanced UI–specific functions that enable interaction between an InstallScript package and the Suite/Advanced UI installation or the Advanced UI installation that is running it. For example, InstallScript includes new functions that let you log InstallScript package information to Suite/Advanced UI and Advanced UI debug logs, set and retrieve Suite/Advanced UI and Advanced UI properties, and pass data from Suite/Advanced UI and Advanced UI installations to the InstallScript package.

To determine whether the InstallScript installation is running as an InstallScript package in a Suite/Advanced UI or Advanced UI installation, use the new SUITE_HOSTED variable in your InstallScript code.

This feature resolves issue IOA-000068090.

New InstallScript Package Type of Condition Check in 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, you can select from a number of different types of checks that you want to be evaluated on target systems. Use the new InstallScript package type of condition check to check target systems for the presence of a product that was installed by a particular InstallScript installation. The condition checks for a particular product code, and it can also check other information, such as the product version.

Dynamic File Link Support for Package Files in Suite/Advanced UI and Advanced UI Projects

When you are adding or configuring an .msi, .msp, or .exe package in an Advanced UI or Suite/Advanced UI project, you can indicate whether the package requires additional files that are located near the package file. For example, if a package that you are adding is an uncompressed .msi package, you may need to include other files—such as .cab files and uncompressed data files in nearby subfolders—along with the package file.

InstallShield now lets you use dynamic links for the additional package files. Dynamic links are useful if the list of additional files that the package requires is likely to change between builds. InstallShield scans the source folder before every build and automatically incorporates any new or changed package files in your release.

InstallShield also lets you define filters that control which additional files InstallShield should include in the dynamic link at build time, and which ones InstallShield should exclude. You can change the order in which InstallShield evaluates any filter criteria that you have defined for a dynamic link. Each time that you build your Advanced UI or Suite/Advanced UI installation, InstallShield includes the appropriate additional files based on the dynamic link's filters.

Previously, only static links could be used for additional files. If the list of additional files changed between builds, it was necessary to manually update the list of package files.

Ability to Give Enhanced Feedback When Validating End-User Input During a Suite/Advanced UI or Advanced UI Installation

Suite/Advanced UI and Advanced UI installations now have support for providing enhanced feedback when validating end-user input at run time. Various interface controls in the Wizard Interface view of a Suite/Advanced UI project and an Advanced UI project have three new subsettings under the existing Text Style setting: Default, Valid, and Invalid. You can configure these subsettings to select different text styles that you want the Suite/Advanced UI or Advanced UI installation to use under different circumstances.

Support for Creating Package Log Files When Launching a Suite/Advanced UI or Advanced UI Installation from the Command Line

When you are configuring the settings for a package in a Suite/Advanced UI or Advanced UI project, you can use the new Enable Logging Support setting to specify whether you want the package to generate a log file if the Suite/Advanced UI or Advanced UI installation is launched from the command line with the new /log command-line parameter. Depending on the type of package (.msi package, .msp package, or some other type of package), you can also configure one or two other settings to specify information such as which log options you want the Suite/Advanced UI or Advanced UI installation to pass to the package when the log file is being created.

The new /log command-line parameter for the Suite/Advanced UI or Advanced UI Setup.exe file lets you specify the path to the directory that contains the package log files. If a path is not specified with the /log parameter, the Suite/Advanced UI or Advanced UI installation creates the package log files in the %TEMP% directory.

The new property ISLogDir in Suite/Advanced UI and Advanced UI installations stores the path to that directory that contains the package log files.

Previously, the only way to enable logging for a Windows Installer–based package in a Suite installation was to use the logging system policy or the MsiLogging property.

Support for 64-Bit Components in InstallScript Installations

InstallScript projects now have support for installing files to WINSYSDIR64 (the InstallScript variable that maps to the 64-bit System32 folder), and for installing registry data to the 64-bit registry locations, without requiring you to modify your InstallScript code. If you have files or registry data that need to be installed to these 64-bit locations, you can add the files and registry data to a component, and select Yes for that component's new 64-Bit Component setting. At run time, the installation automatically disables file system redirection for the component's System32 files, and it prevents redirection for the component's 64-bit registry data.

Previously, to install files to WINSYSDIR64, it was necessary to override the Installing and Installed events for features that contained components that installed to that location. In the Installing event, it was necessary to use the WOW64FSREDIRECTION constant with the Disable function to disable file system redirection; in the Installed event, it was necessary to use WOW64FSREDIRECTION with the Enable function to re-enable file system redirection for other parts of the installation. The same sort of disabling and enabling was necessary for the UnInstalling and UnInstalled events to ensure that those files were removed correctly during uninstallation.

If file system redirection is not disabled when an InstallScript installation installs to WINSYSDIR64, 64-bit Windows automatically redirects the file transfers to the 32-bit System32 folder (SysWOW64).

Also previously, to install registry data to a 64-bit area of the registry, it was necessary to use the InstallScript registry functions to create registry data with REGDB_OPTION_WOW64_64KEY set in REGDB_OPTIONS. Then it was necessary to use REGDB_OPTION_USE_DEFAULT_OPTIONS with REGDB_OPTIONS to re-enable registry redirection for other parts of the installation.

If registry redirection is not disabled when an InstallScript installation installs to a 64-bit registry location (HKEY_LOCAL_MACHINE\Software), 64-bit Windows automatically redirects the registry changes to the equivalent 32-bit registry location (HKEY_LOCAL_MACHINE\Software\Wow6432Node).

The InstallScript log files (.ilg) that InstallScript installations create at run time when installing a product now use a new OPTYPE_FILE64 type to identify files that are installed when file system redirection is disabled. The InstallScript log files use the existing OPTYPE_REGISTRY64 type to identify registry data that are installed when registry redirection is disabled. You can see these OPTYPE_FILE64 and OPTYPE_REGISTRY64 types when viewing an .ilg file in the InstallShield Cabinet and Log File Viewer.

New Locale Type of Condition Check in 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, you can select from a number of different types of checks that you want to be evaluated on target systems. Use the new locale type of condition check to check for matching one or more locale-related settings on target systems.

This feature resolves issue IOA-000067053.

New Wizard Interface Toolbar for Editing the Layout in Suite/Advanced UI and Advanced UI Projects, with Support for Switching Languages in Suite/Advanced UI Projects

If you select a wizard page or secondary window in the Wizard Interface view of a Suite/Advanced UI project, the toolbar that InstallShield shows directly above the wizard interface preview pane includes several different buttons and other controls that let you modify the layout of the selected page or window. InstallShield also displays this toolbar in the Wizard Interface view if you select one or more controls on a wizard page or a secondary window.

The new toolbar has buttons that let you add labels, text boxes, check boxes, and other controls to the installation's user interface. The toolbar also has buttons that let you easily align selected controls, resize them, and position them in relation to each other. In Suite/Advanced UI projects, the Default Languages list in the new toolbar enables you to switch the strings that InstallShield displays on the wizard pages and secondary windows in this view to those in a different language in your project.

Support for Adding Languages to Suite/Advanced UI Projects

The InstallShield Premier edition includes default run-time strings in 35 supported languages. When you add a supported language to a Suite/Advanced UI project, that language is made available in various language-related settings throughout InstallShield. In addition, InstallShield adds translated string entries for that language to your project. The string entries are for the default wizard pages, messages, and other end-user interface elements.

InstallShield now lets you add unsupported languages, beyond the built-in 35 languages, to Suite/Advanced UI projects through the New Language Wizard. An unsupported language is one in which none of the default run-time strings are translated. When you add an unsupported language to a Suite/Advanced UI project, that language is made available in various language-related settings throughout the project. In addition, InstallShield uses the strings from your project's default language as placeholders for the strings in that newly added unsupported language; you can use the String Editor view to provide translated strings for the unsupported languages.

To launch the New Language Wizard in a Suite/Advanced UI project, on the Tools menu, click Add New Language.

Ability to Create and Configure Scheduled Tasks on Target Systems at Run Time

InstallShield has a new Scheduled Tasks view that lets you configure one or more tasks that you want to be created through the Windows task scheduler at run time on target systems. The view lets you specify information such as the file that you want to be launched for a task, as well as the start date and time. The file that you want to be launched can be part of your installation, or it can be a file that is already present on target systems.

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

New FlexNet Connect 13.03 Redistributables Available

InstallShield includes support for FlexNet Connect 13.03 in Basic MSI and InstallScript MSI projects. Use the Update Notifications view in InstallShield to include one of the two FlexNet Connect 13.03 merge modules—one has the Common Software Manager, and the other does not.

Enhancements in InstallShield 2012 Spring Original Release Version (May 2012)

Enhancements to the InstallScript Language for Operating Systems

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

- SYSINFO.WINNT.bWin8—This is a new SYSINFO structure member. If the operating system is Windows 8 or Windows Server 2012, this value is TRUE.
- ISOSL_WIN8—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 or Windows Server 2012.

Automation Interface Enhancement: OSFilter Property Value for Windows 8 and Windows Server 2012

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

eosWin8 = &H4000000 (67108864)—These are for Windows 8 and Windows Server 2012.

In addition, the value for the eosAll constant is now &7D100D0 (131137744); previously, it was &3D100D0 (64028880).

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

Ability to Easily Move Conditions in Suite/Advanced UI and Advanced UI Projects

When you have more than one conditional statement for an exit, detection, eligibility, or feature condition in a Suite/Advanced UI or Advanced UI project, you now can move the conditional statements to reorder conditions or change the hierarchy of a condition tree. For example, if you have a platform conditional statement in a None condition group, and the None condition group is in an All condition group, you can move the platform conditional statement left, so that it is only part of the All condition group.

To move a conditional statement or group, click the new Move Condition button in the setting of the item that you want to move, and then click the appropriate option (Move Up, Move Down, Move Left, or Move Right).

Previously, it was necessary to manually create the new conditional statement in the new location and delete the one in the old location. Or, as an alternative, you could edit the .issuite file in a text editor and change the order of the conditional statements.

New Extension Condition Support and Enhanced Condition Settings in Suite/Advanced UI and Advanced UI Projects

In Suite/Advanced UI and Advanced UI projects, new extension condition functionality is available. This type of condition lets you browse to a C/C++ DLL that you have created to check for your own custom conditions on target systems.

In addition, many of the condition-related settings that are available in Suite/Advanced UI and Advanced UI projects have been enhanced to make it easier to build conditional statements. For example, instead of manually trying to enter a product code, upgrade code, patch code, or other various types of data in any one of several of the condition settings, you can now click a new ellipsis button (...) in the setting; when you do this, InstallShield displays a browse dialog box that lets you browse to and select the appropriate package. Once you have selected a

package, InstallShield enters the appropriate information from that package in the setting of the Suite/Advanced UI or Advanced UI project.

When you are configuring an eligible package condition, you can now select from a list of packages in your Suite/Advanced UI or Advanced UI project instead of having to manually enter the package GUID of the appropriate package.

Enhancements for Configuring Validation and Actions for a Control on a Wizard Page or Window in Suite/Advanced UI and Advanced UI Projects

In Suite/Advanced UI and Advanced UI projects, the Validation and Action settings for various controls on the wizard interface have been enhanced to make it easier to define validation and trigger various actions. These settings now contain drop-down lists of sample statements that you can enter in these settings; these settings are also still text boxes that let you enter statements manually. For example, the Validation setting lets you specify a format that end users must match when they enter a serial number in a control. As an alternative to manually entering the entire validation statement, you can now select the mask type of validation in the Validation setting and then override the default format in the setting.

Similarly, the Action setting lets you define—for example—a print action for a button control; when an end user clicks the Print button, the Print dialog box opens, enabling the end user to print the license agreement. As an alternative to manually entering the action statement, you can now select the print action in the Action setting and then override the file name with the name of the file that you want to be printed.

The drop-down lists in the Validation and Action settings also now include C/C++ DLL files that you have created and added to your project through the Support Files view. This functionality lets you trigger your own validation or your own action for various controls on the wizard interface.

Ability to Use Custom Folder Names for Packages in Advanced UI and Suite/Advanced UI Releases

When you build an Advanced UI or Suite/Advanced UI installation, InstallShield creates a folder for each package that is included in the installation; these folders are created in the same folder that contains the Advanced UI or Suite/Advanced UI setup launcher. By default, the name of each folder is a GUID that InstallShield generates at build time.

The Packages view in InstallShield now lets you override the GUID name with a user-friendly name for each package folder. To enter a custom folder name in this view, find a Package Files folder under the package whose folder name you want to customize. Right-click that folder, click Rename, and enter a new name. If you customize more than one folder name, ensure that each folder name is different.

Previously, InstallShield used a GUID for the name of each folder and did not have support for customizing the names.

This enhancement resolves issue IOA-000067861.

New Formatted Support for Properties Whose Values Reference Other Properties in Advanced UI and Suite/Advanced UI Projects

Each property that is defined in the Property Manager view in an Advanced UI or Suite/Advanced UI project has a new Formatted check box. This new check box lets you indicate whether you want the properties that are referenced in a property's value to be resolved and replaced by their property values at run time.

To replace properties that are enclosed within square brackets (such as [PropertyName]) at run time, select this check box. To leave square brackets and the content within them as is, clear this check box.

This enhancement resolves issue IOB-000061352.

Improved Timing for UAC Prompts of Downloaded Packages that Require Elevation for Suite/Advanced UI and Advanced UI Installations that Have an Invoker Manifest

If you build an Advanced UI or Suite/Advanced UI release with an Invoker manifest, and if Yes is selected in the Require Elevated Privileges setting for any of the installation's packages that need to be downloaded and launched on the target system, the installation triggers the UAC prompt soon after end users click the Install button—before the download occurs.

Previously, the installation triggered the UAC prompt after the download occurred. Thus, if package staging was slow (for example, if the package download took a long time), there could be a big gap between the moment that an end user clicked the Install button and the moment that Windows displayed the UAC prompt for elevation for the package. If an end user did not provide consent or credentials quickly enough, the UAC prompt timed out, and the installation failed.

In some previous cases, using an Administrator manifest was a possible workaround, since the UAC prompt was displayed soon after end users clicked the Install button. However, with this workaround, the entire installation had elevated privileges.

Support for Specifying the Alignment for Text in the Wizard Interface of Suite/Advanced UI and Advanced UI Installations

InstallShield has a number of built-in text styles that define text attributes such as color, size, and font name for the text on the wizard interface of Suite/Advanced UI and Advanced UI projects. You can edit any of the settings for these built-in styles or define your own styles, through the Wizard Interface view in your Advanced UI or Suite/Advanced UI project.

Each built-in or custom text style includes a new Text Alignment setting that lets you select the type of alignment that you want to use for the text in controls that use that particular style.

Enhanced Combo Box Controls for the Wizard Interface of Suite/Advanced UI and Advanced UI Installations

InstallShield lets you add a combo box control to a wizard page or a secondary window in Suite/Advanced UI and Advanced UI projects. This type of control is a combination of two controls by default:

- A box that contains a drop-down list of predefined values
- A text box that lets end users enter a custom value

Previously, if you added a new combo box control, the control contained a drop-down list, but it was not also a text box; that is, end users could not enter a custom value.

To change this control to a drop-down list without the text box (that is, if end users should be able to select a predefined value but not enter a custom value), set the CBS_DROPDOWNLIST style for this control to True.

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

Some enhancements and changes have been made to Aero-formatted wizard pages in Suite/Advanced UI and Advanced UI installations:

- The header and navigation areas of wizard pages are now displayed with the Aero glass effect, or translucency. Previously, only the caption bar of wizard pages were displayed with the Aero glass effect.
- The Aero-formatted wizard pages now use the same layout as Wizard 97–formatted wizard pages. That is, the caption bar on Aero-formatted wizard pages is no longer extra tall; it is the same height as the caption bar on Wizard 97–formatted wizard pages. In addition, the Back button on Aero-formatted wizard pages is now

displayed in the navigation area, which is consistent with the placement on Wizard 97–formatted wizard pages. Previously, the top-left corner of the caption bar in Aero-formatted wizard pages contained the Back button.

Ability to Remove a String Value and Its Identifier from a Setting

When you are entering the value of a setting in one of the views in InstallShield and that value is a text string that can be presented to end users, InstallShield automatically uses a string identifier for that setting. InstallShield places the string identifier in curly brackets before the string value. These types of settings now contain a new "Delete this string reference" button.

If you want to remove the string identifier and its value from a setting, you can now click this new button. The button lets you clear the entry in the setting. Note that if you want to delete the string identifier and its value from your project, you must use the String Editor view.

This enhancement resolves issue IOA-000065831.

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.

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

You can obtain the installations of InstallShield, Standalone Build, and Repackager (which is available with the Premier edition of InstallShield) 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, the Standalone Build, and Repackager.

Installing More than One Edition of InstallShield

Only one edition of InstallShield 2012 Spring—Premier, Professional, or Express—can be installed on a system at a time. In addition, the InstallShield 2012 Spring DIM Editor cannot be installed on the same machine with any edition of InstallShield 2012 Spring.

Installing More than One Version of InstallShield

InstallShield 2012 Spring can coexist on the same machine with other versions of InstallShield.

The InstallShield 2012 Spring 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.

Integrating InstallShield with Visual Studio

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.

Project Upgrade Alerts

The following information describes possible upgrade issues that may occur when you upgrade projects that were created with InstallShield 2012 and earlier to InstallShield 2012 Spring. It also alerts you to possible changes in behavior that you may notice between new InstallShield 2012 Spring projects and projects that are upgraded from InstallShield 2012 or earlier to InstallShield 2012 Spring. For updates to this information, see Knowledge Base article Q204471.

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

If you use InstallShield 2012 Spring to open an project that was created with an earlier version, InstallShield 2012 Spring 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 .772 (for an .ism project) or .2012 (for an .issuite project) before converting it. Delete the .772 or .2012 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 2012 Spring projects in earlier versions of InstallShield.

You can upgrade projects that were created with the following versions of InstallShield to InstallShield 2012 Spring: InstallShield 2012 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 2012 Spring.

New Default Behavior When Launching an Earlier Version or the Same Version of a Suite/Advanced UI Installation on Target Systems

InstallShield now includes two Suite Installed conditions in each Advanced UI and Suite/Advanced UI project by default:

- A new Suite Installed exit condition prevents end users from being able to install the current version of the Advanced UI or Suite/Advanced UI installation over a future newer version of the same Advanced UI or Suite/Advanced UI installation.
- A new Suite Installed mode condition now causes the Advanced UI or Suite/Advanced UI installation to run in first-time installation mode if end users install a new version of the Advanced UI or Suite/Advanced UI installation over an older version of the same Advanced UI or Suite/Advanced UI installation.

These new default conditions are available in all new Suite/Advanced UI projects. If you upgrade an InstallShield 2012 Suite project to InstallShield 2012 Spring, InstallShield automatically adds these default conditions to the project.

Previously, if an end user installed a particular version of a Suite installation on a target system on which a newer version of the Suite was already installed, the installation could permit the end user to install the older Suite version over the newer version. In addition, if an end user installed a particular version of a Suite installation on a

target system on which the same version of the Suite was already installed, the Suite installation could run in first-time installation mode. Furthermore, if an end user installed a new version of a Suite installation on a target system on which an older version of the Suite was already installed, the Suite installation could run in maintenance mode.

Automation Interface Changes

If you use the automation interface with InstallShield or the Standalone Build, update your existing code to reflect the new ProgID: IswiAuto19.ISWiProject. The Standalone Automation Interface uses the same ISWiAutomation19.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 ISWiAutomation19.dll file that is registered is the one that is used.

Changes to the Default Behavior of New Combo Boxes in the Wizard Interface of Suite/Advanced UI Projects

If you create a new combo box control on a wizard page or a secondary window in a Suite/Advanced UI project in InstallShield 2012 Spring, the control is a box that contains a drop-down list of predefined values. The box is also a text box that lets end users enter a custom value. Previously, if you added a new combo box control in InstallShield 2012, the control contained a drop-down list, but it was not also a text box; that is, end users could not enter a custom value.

If you upgrade the project from InstallShield 2012 to InstallShield 2012 Spring, and if the Suite/Advanced UI wizard interface includes a combo box, the combo box is left as a drop-down list of predefined values, but it is not also a text box. To change the control to a drop-down list with the text box, set the CBS_DROPDOWNLIST style for this control to False.

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 2012, you can still do so. For instructions, see Knowledge Base article Q200884.

Resolved Issues in InstallShield 2012 Spring Original Release Version (May 2012)

1-12NWKD, IOA-000065414

The sample InstallScript code that is in the "Specifying a Component's Destination from the Script" help topic has been corrected. The ComponentSetTarget call has been replaced with a call to FeatureSetTarget.

IOA-000057809 (InstallShield Prerequisite Editor)

You can now include files that have mixed character sets in their file names in InstallShield prerequisites that you create in the InstallShield Prerequisite Editor. Previously, the editor used question marks in place of Unicode characters in the .prq file.

IOA-000061630 (InstallScript MSI)

The InstallScript variable IFX_INSTALLED_VERSION is now initialized as expected during an InstallScript MSI installation in which the InstallScript engine is used as an embedded UI handler and the target system has an earlier version of the product installed. Previously, the variable was initialized to a null string.

IOA-000065376, IOA-000065727

If you use MSBuild to build a solution that contains an InstallShield project, a build error about a missing InstallShield.targets file no longer occurs. In InstallShield 2012, this error occurred because the MSBuild support in InstallShield was installed to the path for an earlier version of InstallShield.

IOA-000065578

The sample InstallScript code that is in the "StrGetTokens Example" help topic has been corrected. The following code is used:

if (StrGetTokens (listID, svSearchPath, ";") < 0) then

Previously, the code erroneously used a greater than symbol instead of the less than symbol.

IOA-000065602 (Basic MSI, InstallScript MSI, Suite/Advanced UI)

If you build a compressed Setup.exe file for a Basic MSI, InstallScript MSI, or Suite/Advanced UI project with an .msi file name that contains Unicode characters, a run-time error no longer occurs when Setup.exe attempts to extract the .msi file to a temporary location.

IOA-000065717

If you use the condition "A registry entry has a specified value" or the condition "A registry entry has a specific version value" in an InstallShield prerequisite, the condition is now evaluated properly at run time. Previously in some cases, the condition evaluated incorrectly for some registry comparisons that checked for a value that was less than or greater than a specific number.

IOA-000065830 (InstallShield Prerequisite Editor)

Opening and then closing the InstallShield Prerequisite Editor no longer triggers the Program Compatibility Assistant dialog box to open and indicate that the program may not have installed correctly.

IOA-000065893 (Basic MSI)

The German version of the multiple instance run-time dialog no longer uses the same keyboard shortcuts for one of the radio buttons and the Next button.

IOA-000065939 (Basic MSI)

The Korean version of the multiple instance run-time dialog now has the Korean translation instead of the English word "Version."

IOA-000065940 (Basic MSI)

The Polish version of the multiple instance run-time dialog now has the word "Wersja" instead of "Wersji" for the word "Version."

IOA-000066021 (Basic MSI, InstallScript MSI)

If you create a major upgrade for an earlier installation that includes a chained package, if both the base installation and the major upgrade are created in InstallShield 2012 Spring, and if an end user tries to install the

base and then the major upgrade, the removal of the base before the major upgrade occurs no longer fails and rolls back. As a workaround, the base package should have the conditional statement And Not UPGRADINGPRODUCTCODE added to the install and uninstall conditions for the chained package.

Note that a chained package cannot be run during the removal of a base package via major upgrade. However, each chained package can remove its own previous version using a separate major upgrade.

IOA-000066096 (Basic MSI, DIM, InstallScript, InstallScript MSI)

The Schema Version setting on the General tab for a SQL script in the SQL Scripts view now lets you enter a schema version that contains a maximum of 99999 for each of four version fields. Previously, the setting let you enter only three fields; the maximum for the first and third fields was 9999, and the maximum for the second was 99999.

IOA-000066181, IOA-000068092 (Basic MSI, InstallScript, InstallScript MSI)

If you use the InstallScript Debugger to debug an InstallScript function that uses variable arguments or local array variables, the debugger no longer encounters various exceptions or ignores breakpoints. Previously, debugging these types of functions led to unexpected results.

IOA-000066216 (Standalone Build)

If you use multiple instances of the Standalone Build, you can now run multiple builds in parallel. Previously in some cases, the Standalone Build encountered fatal error -5092 at built time.

IOA-000066274 (Suite/Advanced UI)

Visual Studio no longer crashes when you add a Suite/Advanced UI project to a Visual Studio solution.

IOA-000066278

The OpenFileMode help topic in the InstallShield documentation has been corrected. It now states that you can use ListWriteToFileEx instead of ListWriteToFile with appropriate options.

IOA-000066279

The InstallScript function WriteLine now handles double-byte characters as double-byte characters when writing to an ANSI file; previously, it handled them as single-byte characters.

IOA-000066853, IOA-000066999 (Suite/Advanced UI)

If a Suite/Advanced UI installation contains a 64-bit compressed .msi package, the package is now run on 64-bit systems if appropriate. Previously, the installation failed, and the debug log showed code 800700e9 and contained a message indicating that the installation timed out waiting for the 64-bit elevated proxy to respond.

IOA-000066854 (Basic MSI, DIM, InstallScript, InstallScript MSI)

If your installation runs a SQL script that was included in the SQL Scripts view of your project, and if the last line of the SQL script contains a batch separator but no line ending, the installation no longer encounters a run-time error about incorrect syntax near the batch separator.

IOA-000067051 (InstallShield Prerequisite Editor)

When you use the Files to Include tab in the InstallShield Prerequisite Editor to add files to a prerequisite, the editor now uses path variables properly for the Path to Local File column. Previously in some cases, the path variables were used incorrectly. For example, if the file being added was C:\Windows\System32 Scratch\MyFile.exe, the editor showed the path to the local file as <SystemFolder> Scratch\MyFile.exe, and the path could not be resolved.

IOA-000067135 (Basic MSI, InstallScript MSI)

The Swedish value of string identifier IDS_ERROR_27502 has been corrected. Previously, one of the placeholder variables was used twice in the string value.

IOA-000067153 (Suite/Advanced UI)

If you use a condition that checks for a file in a 64-bit location (such as [ProgramFiles64Folder], or a subfolder of this location) in a Suite/Advanced UI project, the Suite/Advanced UI installation now evaluates the condition properly at run time. Previously, a backslash was missing from the path.

IOA-000067270

In the "Product Version Numbers in InstallScript and InstallScript Object Projects " help topic, the sample RegDBGetKeyValueEx call has been corrected; the registry path has double backslashes, instead of forward slashes.

IOA-000067321 (Basic MSI, InstallScript, InstallScript MSI)

The InstallScript Debugger now makes the breakpoints that were set in a previous session available in a new debugging session.

IOA-000067333 (Basic MSI, InstallScript MSI, Suite/Advanced UI)

When an installation generates a debug log file on a Japanese system, the log file now contains appropriate line feed characters.

IOA-000067508, IOA-000069594 (Suite/Advanced UI)

When a Suite/Advanced UI installation is run in maintenance mode, the installation no longer misrepresents and mishandles the feature states. In addition, maintenance mode now enables end users to select and fully install a feature that is only partially installed, or to uninstall the product.

IOA-000067770 (Basic MSI, InstallScript MSI, Merge Module, MSI Database)

An ISICE02 validation error no longer causes an unhandled exception.

IOA-000067816 (Suite/Advanced UI)

Moving features under other features and then moving the parent features around no longer causes unexpected additions and removals of features.

IOA-000068396 (InstallScript)

The hyperlink in the inline help of the Objects view has been updated. It now points to the Flexera Software Product and License Center, where you can obtain installations for the InstallScript objects that you can add to InstallScript projects.

IOA-000068428 (Basic MSI, InstallScript, InstallScript MSI)

The InstallShield dependency scanners can now detect and find dependencies that are non-PE files, such as config files.

IOA-000068505 (InstallScript)

If you insert a SQL script in the SQL Scripts view and the name of the script file has a hyphen, InstallShield no longer crashes.

IOA-000068511 (Basic MSI, InstallScript, InstallScript MSI)

If you select Yes for the AspEnableParentPaths setting of an IIS application or virtual directory now, the installation now selects the Enable Parent Paths check box for the application or virtual directory on the target system at run time. Previously, the check box remained cleared.

IOA-000068513 (Basic MSI)

If you leave software identification tagging enabled but not configured, and if you enter digital signature information for a release, InstallShield no longer displays a build error about failing to sign the release.

IOA-000068704 (Basic MSI, Merge Module)

If you enter a value in the Max. Length setting for an edit field control or a combo box control on a dialog box, InstallShield now retains the value. Previously, the value was not retained.

IOA-000068706 (Basic MSI)

If an end user runs an installation in maintenance mode and chooses to install a feature that contains a feature prerequisite that has not yet been installed, the feature prerequisite can now be installed without prompting the end user to locate the Setup.exe file of the main installation. Previously in some cases, the Setup.exe file could not be found, and the installation prompted the end user to browse to a file with a truncated file name of the appropriate file.

IOA-000068715 (Basic MSI)

If you build a release in which you are using release flags to exclude a feature prerequisite, the resulting built installation does not attempt to install the excluded feature prerequisite. Previously in some cases, the installation ran into an infinite loop because it tried to install the excluded feature prerequisite.

IOA-000068802 (Suite/Advanced UI)

If you specify a postbuild event for a Suite/Advanced UI release, the event now runs after InstallShield has built the Setup.exe file. Previously, the event was triggered before the Setup.exe file was built.

IOA-000068835 (Basic MSI, InstallScript, InstallScript MSI)

If you create and run an installation that should install a nested application tree for applications that contain nested virtual directories, the nested virtual directories are now available in Internet Information Services Manager 7 and later. Previously in this scenario, double-clicking a nested virtual directory in the IIS Manager on a target system triggered an error about the virtual directory not existing.

IOA-000069099

InstallShield now includes the predefined constants ISOSL_WIN7_SERVER2008R2 and ISOSL_WINVISTA_SERVER2008 in the auto completion pop-up list when you start to type either of these constants in a script file in the InstallScript view.

IOA-000069199

If an end user silently runs an installation that attempts to modify a read-only XML file whose changes were configured in the XML File Changes view, the installation now aborts. Previously, the installation got caught in an infinite loop trying to modify the file.

IOA-000069311 (InstallScript)

If you enable font registration in an InstallScript project and include a font file (.ttf) for which the file is marked to be permanent and registered, the product can now be uninstalled from a target system. Previously, logging was

left disabled after registering the font on target systems, and this prevented the installation from logging the uninstall key.

IOA-000069502

The descriptions of MyOnBegin and MyOnMoving in the "Creating and Scheduling InstallScript Custom Actions that Call InstallScript Event Handlers for Basic MSI Projects" help topic have been corrected.

IOA-000069577 (Basic MSI, InstallScript MSI)

If your installation creates an IIS application pool identity that was configured in the Internet Information view, the installation no longer includes the application pool's identity password in the Windows Installer log file when the password is stored as the value of a Windows Installer property.

IOA-000069703 (Suite/Advanced UI)

The final exit code of a Suite/Advanced UI installation now reflects the most severe error, if applicable. Previously, the final exit code of the Suite/Advanced UI installation was the exit code of the final package that the installation processed.

IOA-000069707 (Virtualization)

If you configure default values for various virtual settings in the Settings.xml file that is installed with InstallShield, and if there are no corresponding values in the ISVirtualPackage table of the project, the InstallShield virtualization assistants now use those default values. This applies to the App-V support in InstallShield and AdminStudio, as well as the ThinApp and XenApp support in AdminStudio.

IOA-000069733 (InstallScript MSI)

The message for Windows Installer run-time error 1603 no longer displays question marks in place of text when an InstallScript MSI installation encounters this error and an Asian language is used for the run-time language.

IOB-000061211 (Suite/Advanced UI)

If a Suite/Advanced UI project includes a long EULA or other document that is used in a rich text control on a wizard page such as the LicenseAgreement wizard page, the end of the text is no longer cut off at run time.

IOC-000087762

The Slovak version of the DiskSpaceRequirements run-time dialog now has the correct translation.

IOC-000087763

In the Slovak language file 0x041b.ini, the names of the languages that previously started with lowercase letters now start with uppercase letters.

IOC-000087885 (Basic MSI, InstallScript MSI)

If you use the ellipsis button (...) in the INSTALLDIR setting in the General Information view to select a predefined folder, InstallShield uses square brackets around the name of the directory property. In InstallShield 2012, InstallShield omitted the square brackets, and it was necessary to manually enter them.

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 XP
Windows Server 2003
Windows Vista
Windows Server 2008
Windows 7
Windows Server 2008 R2
Windows 8
Windows Server 2012

Browser

Microsoft Internet Explorer 6

Privileges

Administrative privileges on the system

Mouse

Microsoft IntelliMouse or other compatible pointing device

For Target Systems (Desktop Computers)

Target systems must meet the following minimum operating system requirement:
Windows 2000
Windows XP
Windows Server 2003
Windows Vista
Windows Server 2008
Windows 7

Windows Server 2008 R2
Windows 8
Windows Server 2012

For Target Systems (Mobile Devices)

InstallShield includes support for adding mobile device installations to desktop installations that use Microsoft Windows Mobile Device Center or Microsoft ActiveSync to transfer files to a mobile device.

InstallShield also includes support for straight-to-device installations that do not use Windows Mobile Device Center, ActiveSync, or any other desktop component.

For an overview of the different options that InstallShield supports, see "Creating Installations for Mobile Devices" in the InstallShield Help Library.

Windows Mobile Device Requirements

InstallShield supports many Windows Mobile platforms and processors. The Windows Mobile platforms are:

- Windows Mobile 6.x Professional and Classic
- Windows Mobile 6.x Standard
- Windows Embedded CE 6.x
- Windows Mobile 5.0 for Pocket PC
- Windows Mobile 5.0 for Smartphone
- Windows CE .NET 5.0
- Windows CE .NET 4.x
- Pocket PC 2003
- Pocket PC 2002
- Pocket PC
- Palm-size PC 2.11
- Palm-size PC 2.01
- Handheld PC 2000
- Handheld PC Pro
- Handheld PC 2.0
- Smartphone 2003
- Smartphone 2002

Note that if a platform is not included in the list, it does not mean InstallShield does not support it. It simply means that you cannot set conditions for that specific platform by default. To add support for additional platforms or to change the conditions for targeting a specific platform, you can modify the Settings.xml file that is installed with InstallShield. For more information, see "Modifying the List of Available Windows Mobile Platforms or their Associated Settings" in the InstallShield Help Library.

InstallShield includes support for the following Windows Mobile processors:

- ARM920
- ARM820
- ARM720

- Common Executable Format
- Hitachi SH4
- Hitachi SH3E
- Hitachi SH3
- i686
- i586
- i486
- MIPS R4000
- MIPS R3000
- MIPS R2000
- SHx SH4
- SHx SH3
- StrongARM-XScale

Palm OS Device Requirements

InstallShield supports Palm OS 3.5 and later.

Desktop Requirements for Windows Mobile Device Installations

Requirements for the desktop computers that are used to install applications on Windows Mobile devices are:

- Microsoft ActiveSync 3.x or later on Windows XP (ActiveSync 4.x is required for Windows Mobile 5.x or later devices)
- Microsoft Windows Mobile Device Center on Windows Vista
- Administrative privileges

Desktop Requirements for Palm OS Device Installations

Palm HotSync is required for the desktop computers that are used to install applications on Palm OS devices.

Known Issues

For a list of known issues, see Knowledge Base article Q204469.