InstallShield
www.installshield.com

Get the latest versions of InstallShield help files

For the latest versions of InstallShield help files, visit our documentation Web page.

InstallShield
www.installshield.com

Visit the InstallShield Knowledge Base

For answers to many commonly asked questions and new information about InstallShield that may not appear in the documentation, visit the InstallShield Knowledge Base.

Overview

InstallShield provides the most powerful and versatile setup creation technology available. The new InstallShield integrated development environment (IDE) makes creating world-class setups a snap. InstallShield makes all aspects of setup creation, including file transfer, the core of any setup, easier and more intuitive than ever before.

This Getting Started document describes the building blocks of setups and walks you through the creation of a complete setup. In just a few minutes, you will able to create setups that include uninstallation functionality and Start Programs menu access, and build disk images optimized for your distribution media.

{button ,JI(`GETSTART.HLP',`InstallShield_Free_Edition')} InstallShield Free vs. Professional Edition

InstallShield
www.installshield.com

InstallShield Free vs. Professional Edition

InstallShield Free Edition is a limited version of InstallShield Professional Edition. In this topic we summarize the differences between the Free Edition and the Professional Edition. Throughout the InstallShield documentation, features that are limited or unavailable in the Free Edition are marked with this graphic: professional edition only. Click the graphic for more information on how that feature differs in the Free Edition and the Professional Edition.

The differences between InstallShield Free Edition and InstallShield Professional Edition are these:

Technical support
Registered users of InstallShield Free Edition get free technical support only for 30 days from the date of their first inquiry. Registered users of InstallShield Professional Edition (or InstallShield International) get 30 days of free support from the date of the response to their first inquiry—and the opportunity to purchase additional support.

16-bit Windows setups
InstallShield Free Edition does not create setups for 16-bit Windows platforms.

PackageForTheWeb
InstallShield Free Edition does not include PackageForTheWeb, which lets you package your setup into a self-extracting .exe or .cab file for distribution via floppy disk, CD-ROM, or Internet. Users of InstallShield Professional Edition (or InstallShield International) can find PackageForTheWeb on their InstallShield CD-ROM or contact sales@installshield.com.

Components
InstallShield Free Edition does not let you create more than five components or let you create subcomponents. InstallShield Professional Edition lets you create any number of components and lets you create subcomponents.

Component Wizard
InstallShield Free Edition does not have the Component Wizard.

File groups
InstallShield Free Edition does not let you create more than eight file groups. InstallShield Professional Edition lets you create any number of file groups.

Compression
InstallShield Professional Edition offers enhanced compression. InstallShield v5.0 Free Edition compression is the same as that of InstallShield3.

DLL functions
InstallShield Professional Edition allows prototyping of DLL functions. InstallShield Free Edition offers only the limited CallDLLFx function.

Initialization dialog
InstallShield Free Edition does not let you modify or hide the initialization dialog.

Media build report
InstallShield Free Edition does not generate a media build report.

End-user dialog box bitmap
InstallShield Free Edition does not let you replace or modify the bitmap that appears in many InstallShield end-user dialog boxes.

InstallScript functions
InstallShield Free Edition does not include the following InstallScript functions:

Extensibility functions	AppCommand DoInstall LaunchApp LaunchAppAndWait UnUseDLL UseDLL
Multimedia functions	PlayMMedia SetDisplayEffect
.Ini file functions	AddProfString GetProfInt

	GetProfString ReplaceProfString WriteProfString
File functions	FileCompare FileDeleteLine FileGrep FileInsertLine
Binary I/O functions	CopyBytes ReadBytes SeekBytes WriteBytes
Remote registry functions	RegDBConnectRegistry RegDBDisConnectRegistry
Component filtering functions	ComponentFilterLanguage ComponentFilterOS
Other functions	FindWindow ParsePath SendMessage StrGetTokens

InstallShield Free Edition does not include any custom dialog functions.

InstallShield Free Edition includes only the following Sd dialog functions:

SdAskDestPath
SdComponentDialog2
SdConfirmNewDir
SdFinish
SdFinishReboot
SdLicense
SdProductName
SdRegisterUserEx
SdSelectFolder
SdSetupTypeEx
SdShowInfoList
SdStartCopy
SdWelcome

InstallShield Free Edition does not include the DialogSetInfo function.

Click the "Professional Edition Only" graphic for more information on how a feature differs in InstallShield Free Edition and InstallShield Professional Edition.

Setup building blocks

InstallShield lets you design a setup using files, file groups, components, and setup types as the building blocks.

When creating your setup with InstallShield, you first bundle files into file groups. Then, you link file groups to components and subcomponents. Finally, you associate components with setup types. At run time, your customer simply selects the components and subcomponents he or she wishes to install.

Files and file groups

The bottom line in any setup is the transfer of files to the target system. Application files naturally fall into groups with similar characteristics: system DLLs, application executable files, help files, and so on. Any such group of files is a candidate for becoming a file group.

File groups are comprised of files with similar characteristics, such as purpose/function, shared or not shared, potentially locked or not locked, compressed or uncompressed, target platform operating system the files support, and languages. It may be useful to think of files and file groups as your (setup author) view of the application.

InstallShield
www.installshield.com

Components and subcomponents

Components and subcomponents give you almost limitless flexibility to package your application and related software accessories for setup. It may be useful to think of components and subcomponents as your customer's view of your application.

If your setup installs a suite of products, you can make each product in the suite a component: for example, Word Processor, Spreadsheet Program, Database Program, and Drawing Program. Each component (product) could then have subcomponents allowing the user to install optional pieces of the program: help files, tutorials, optional support files such as graphics libraries, and so on. With InstallShield, you are free to take advantage of the component-subcomponent relationship to achieve any setup configuration you desire.

Setup types

Setup types give your customer options as to which parts of your application(s) to install. Setup type options can save customers time and disk space. For example, the customer may opt not to install a graphics library or a set of tutorials. In addition, setup types allow you to simplify setup for the customer by shielding him or her from what may be perplexing situations. You might, for example, create setup types called "Client Setup - Save Space" and "Client Setup - Save Time." These setup types could allow the customer to get the optimal setup without having to make difficult decisions about which components to install on the client machine and which to install on the server.

Use the Project Wizard

Follow the steps below to create a professional-looking setup, including uninstallation functionality, in 15 minutes or less. You can use the browse buttons to move through the steps.

1. Take advantage of the Project Wizard.
2. Assign files to file groups.
3. Assign file groups to components.
4. Place your application icon on the Start Programs menu.
5. Build your disk images.
6. Run your new setup.

Take advantage of the Project Wizard

The Project Wizard lets you quickly create a runnable setup, which you can then customize. The Project Wizard creates a setup script and makes IDE settings for operating systems, alnguages, setup types, components, and file groups.

1. If you have a project open, close it by selecting Close from the File menu.
2. Double-click the Project Wizard icon in the Projects window. The InstallShield Project Wizard runs.
3. Enter some basic information about your application. In the Welcome panel, do the following (do not include the quotes):
   a) Enter "Test App" in the Application Name field.
   b) Select "Microsoft Visual C++ 5.0" in the Development Environment field.
   c) Select "Database Application" in the Application Type field.
   d) Leave the Application Version field as is.
   e) In the Application Executable field, browse and select Notepad.exe in your Windows folder.
   f) Click Next.
4. Select the dialog boxes you want to display during your setup. In the Choose Dialogs panel, select the Welcome and Choose Destination Location dialogs. Make sure all other dialogs are unchecked. Click Next.
5. Select the operating systems on which your setup runs. In the Choose Target Platforms panel, click Windows 95 in the Operating System list box. Click Next.
6. Select the languages your setup supports. Select English in the Specify Languages panel and click Next.
7. Select the setup types you want to offer your end user. In the Specify Setup Type panel, click in the white space in the Setup Type list to deselect all setup type choices. Click Next.
8. Specify the components you want to define in your setup. In the Specify Components panel, delete all component names except Program Files. You should see only the Program Files component when you are done. Click Next.
9. Specify the file groups you want to define in your setup. In the Specify File Groups panel, delete all file group names except Program Executable Files. You should see only the Program Executable Files file group when you are done. Click Next.
10. The Summary panel summarizes your choices and settings thus far. Click Finish. InstallShield will create your setup project and open the Project workspace.

Congratulations! You just created an InstallShield setup project. With just a few more mouse clicks and a single addition to your script, your setup will add an icon to the Start Programs menu, make an uninstallation entry in the Add/Remove Programs window in Control Panel compile, run, and install files onto a Windows 95 system!

Assign files to file groups

File groups are comprised of logically related files. In this step, you include the file Notepad.exe in the Program Executable Files file group.

1. In the project workspace, click the File Groups pane tab.
2. In the File Groups pane, double-click the Program Executable Files folder icon and click its Links subfolder. The File Group Links window opens.
3. Right mouse click in the File Group Links window. A popup menu appears.
4. From the popup menu, select Insert Files... The Insert File Links to FileGroup dialog opens.
5. Select Notepad.exe from your Windows folder. Click the Open button. The dialog closes, and the selected file appears in the File Group Links window.

Assign file groups to components

Components are the fundamental units of a setup's file transfer. Components are associated with file groups. In this step, you assign the Program Executable Files file group to the Program Files component.

1. In the project workspace, click the Components pane tab.
2. Double-click the Program Files folder icon. The Component Properties window opens.
3. In the Component Properties window, double-click the Included File Groups item. The Included File Groups property page opens.
4. Click the Add... button. The Add File Group dialog opens.
5. In the Add File Group dialog, select Program Executable Files in the File Group Name list box.
6. Click the OK button. The Add File Group dialog closes.
7. Click the OK button. The Included File Groups property page closes.

InstallShield
www.installshield.com

Build your disk images

The Media Build Wizard lets you build disk image folders appropriate for any distribution medium, and it lets you specify the languages and operating systems for that build.

1. In the project workspace, click the Media pane tab.
2. Double-click the Media Build Wizard icon in the Media pane. The Media Build Wizard opens.
3. Enter "Test App" in the Media Name field of the Media Name panel (do not include the quotation marks). Click the Next button.
4. In the Type field of the Disk Type panel, select 3.5" Diskettes - 1.44 Mbytes. Click the Next button.
5. Select Full Build in the Build Type group box of the Build Type panel.
   professional edition only Uncheck the Review Report Before Build checkbox. Click the Next button.
6. The purpose of tag files is to mark each disk of a multi-disk distribution; you can place application information in them if you wish. In the Tag File panel, do the following (do not include quotation marks):
   a) Enter "Test Company" in the Company Name field.
   b) Enter "Test App" in the Application Name field.
   c) Select "Development Tool" in the Product Category field.
   d) Leave the Misc field blank. Click the Next button.
7. The Summary panel displays a summary of the choices and entries you made. Click the Finish button.
8. When the MediaBuild Wizard finishes building your disk images, the Finish button will come back into focus. Click Finish and InstallShield will add your new disk images (and report and log files) to the Media tree in the Media pane.

Only InstallShield Professional Edition generates build reports.

Place your application icon on the Start Programs menu

Your setup is controlled by the script. In this step, you add script code to place your application's icon on the Start Programs menu.

First find the SetupFolders() function definition, which is where you will add the code.

1. In the project workspace, click the Scripts pane tab.
2. Setup.rul should be visible in the Script Editor window in the client area. If it is not, click Setup.rul in the Scripts files tree.
3. Find the SetupFolders() function definition in Setup.rul.
   a) From the Edit menu, select Find... . The Find dialog box opens.
   b) In the Find dialog box, type "function SetupFolders" in the "Find what" box and click the Find Next button.
   c) Click the Cancel button.

Now add the code to place your application's icon on the Start Programs menu: declare a string variable, set the variable equal to the path to your application, and pass the variable to the LongPathToQuote and AddFolderIcon functions.

4. Make the following variable declaration in the function header (just above the begin keyword):
   ```
   STRING svPath;
   ```
5. Place your cursor in the function definition below the TODO comment block.
6. Enter the following text to set the variable equal to the path to your application:
   ```
   svPath = TARGETDIR ^ "Notepad.exe";
   ```
7. Press Enter to begin a new line.
8. Right click in the new line. A popup menu appears.
9. From the popup menu, select Function Wizard... . The Function Wizard opens to let you paste function calls in your script.
10. In the Function Wizard, select "Long filename" in the Function Category list box.
11. In the Function Name list box, select LongPathToQuote. This function places double quotation marks around long filenames, as required by the AddFolderIcon function.

    To learn more about the LongPathToQuote function, click the Help button in the Function Wizard.

12. Click the Next button. The next panel of the Function Wizard opens.
13. Click the Finish button. The function call is pasted into your script.
14. Press Enter to begin a new line.
15. Right click in the new line.
16. From the popup menu, select Function Wizard... .
17. In the Function Wizard, select "Shell" in the Function Category list box.
18. In the Function Name list box, select AddFolderIcon. This function adds an icon to the specified folder.
19. Click the Next button. The next panel of the Function Wizard opens.
20. Make the following entries in the edit and list boxes:

szProgramFolder	""
szItemName	"Test App"
szCommandLine	svPath

szWorkingDir	""
szIconPath	""
nIcon	0
szShortCutKey	""
nFlag	REPLACE

Include all quotation marks.

21. Click the Finish button. The function call is pasted into your script.

To get help for a function in your script, place your cursor in the function name and press F1.

22. Choose Save from the File menu to save your project.

Run your new setup

From the Build menu, select Run Setup. Your setup will run.

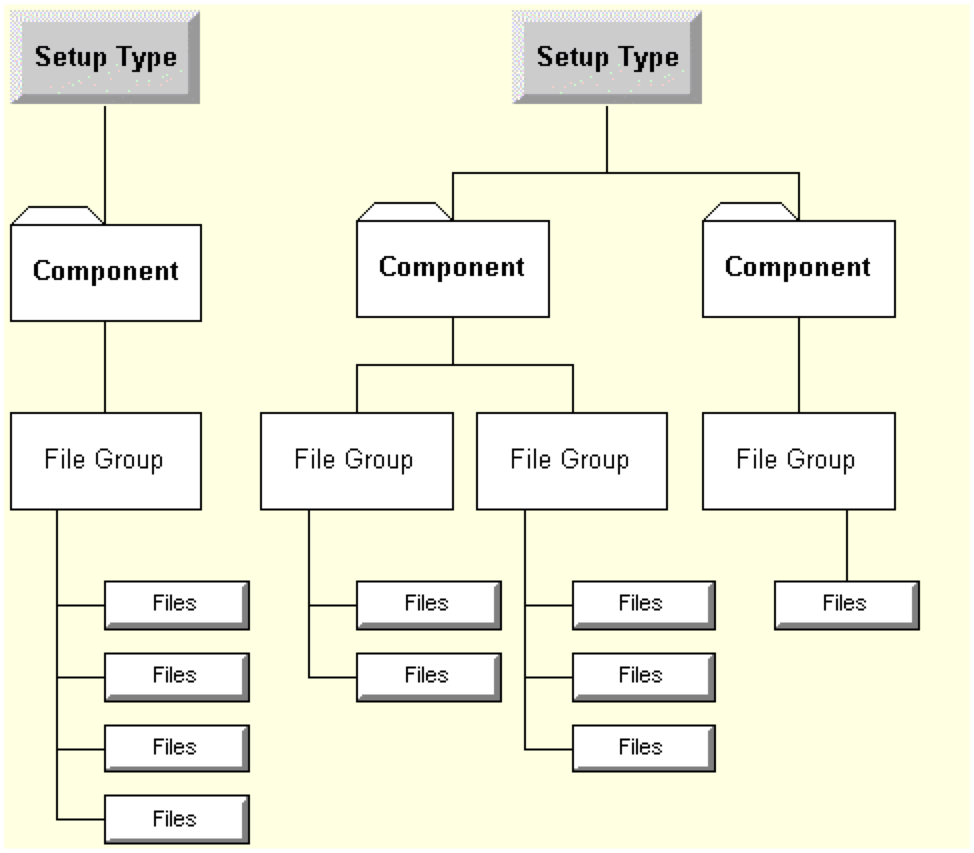
When the setup is complete, you will be able to run Test App (Notepad) from the Start Programs menu. To uninstall Test App do the following:

1. In the Control Panel, double-click Add/Remove Programs, or click here.
2. From the list box, select Your Product Name.
3. Click the Add/Remove button.

InstallShield help is attempting to send you to the InstallShield Web site (www.installshield.com). If your preferred Web browser is not launched, click here , then restart InstallShield help.

InstallShield help found Setbrows.exe but could not run it. Double-click the <InstallShield location>\Program\Setbrows.exe icon.

InstallShield help could not find Setbrows.exe. You may have deleted this file; check to see if Setbrows.exe is in your <InstallShield location>\Program folder.


Setup Type
Setup Type
Component
Component
Component
File Group
File Group
File Group
File Group
Files

InstallShield
www.installshield.com

Use an InstallShield template

Follow the steps below to create a professional-looking setup, including uninstallation functionality, in 15 minutes or less. You can use the browse buttons to move through the steps.

1. Copy a template as a new setup project.
2. Modify the file groups.
3. Modify the components.
4. Modify the Start Programs menu entry.
5. Add function calls.
6. Build your disk images.
7. Run your new setup.

Copy a template as a new setup project

1. If you have a project open, close it by selecting Close from the File menu.
2. From the File menu, select New. The New dialog box opens.
3. In the New dialog box, click the Template tab.
4. Double-click the Template One icon. A project titled "Template One-1" is created and opened.

Before going on to modify the project, see what the setup does in its unmodified state.

1. From the Build menu, select Run Setup. Observe what happens.
2. Uninstall the files.
    a) In the Control Panel, double-click Add/Remove Programs, or click here .

b) From the list box, select Template One.
c) Click the Add/Remove button.

Modify the file groups

File groups are comprised of logically related files. In this step, you delete the file Notepad.exe from the Program Executable Files file group and add the file Write.exe.

1. In the project workspace, click the File Groups pane tab.
2. Click the plus sign (+) next to the Program Executable Files folder. A Links icon appears below the folder.
3. Click the Links icon. The File Group Links window opens.
4. In the File Group Links window, place the mouse pointer over Myapp.exe and click the right mouse button. A popup menu appears.
5. From the popup menu, select Delete. Click the Yes button in the confirmation dialog box.
6. Click the right mouse button again. From the popup menu, select Insert Files... . The Insert File Links to File Group dialog box opens.
7. In the Insert File Links to File Group dialog box, select Write.exe from your Windows folder, and click the Open button. Write.exe appears in the File Group Links window.

Modify the components

Components are the fundamental units of a setup's file transfer. Components are associated with file groups. In this step, you remove the Program DLL Files file group from the Program Files component.

1. In the project workspace, click the Components pane tab.
2. Click the Program Component folder. The Component Properties window opens.
3. In the Component Properties window, double-click Included File Groups. The Included File Groups property page opens.
4. In the Included File Groups property page, select Program DLL Files and click the Remove button, then click the OK button.

Modify the Start Programs menu entry

First notice how the Start Programs menu entry is specified.

1. In the project workspace, click the Scripts pane tab. The script editor window opens with the script file Setup.rul as its contents.
2. In Setup.rul, go to the InstallProgramItems block.
   a) From the Edit menu, select Find... . The Find dialog box opens.
   b) In the Find dialog box, type "InstallProgramItems" in the "Find what" box and click the Find Next button.
   c) Click the Cancel button.

Note that the program folder name is specified by the argument @DEFAULT_FOLDER_NAME. The @ signifies a reference to a string resource in the project's string table. Now edit this resource.

1. In the project workspace, click the Resources tab.
2. Click the English icon. The Strings-English window opens.
3. In the Strings-English window, double-click "DEFAULT_FOLDER_NAME". The Add/Modify String Entry dialog box opens.
4. In the Add/Modify String Entry dialog box, change the value of the string resource from "Template Application" to "New Template Application" and click the OK button.

Now that you know your way around the script editor, change the script's references to "MYAPP.EXE" to "Write.exe".

1. In the project workspace, click the Scripts pane tab.
2. From the Edit menu, select Replace... . The Replace dialog box opens.
3. In the Replace dialog box, enter "MYAPP.EXE" in the "Find what" edit box.
4. Enter "Write.exe" in the "Replace with" edit box.
5. Under "Replace in", click the "Whole file" radio button.
6. Click the Replace All button.
7. Click the Cancel button.

Add function calls

In this step, you add script code to display a message to your end user while your setup checks system requirements.

1. With the script editor window active, select Find... from the Edit menu. The Find dialog box opens.
2. In the Find dialog box, type "CheckRequirements();" in the "Find what" box.
3. Click the Find Next button.
4. Insert a new line before the `CheckRequirements();` line. Place the cursor at the beginning of the `CheckRequirements();` line and press Enter.
5. Right click in the new line. A popup menu appears.
6. From the popup menu, select Function Wizard... . The Function Wizard opens to let you paste function calls in your script.
7. In the Function Wizard, select "Sd dialog box" in the Function Category list box.
8. In the Function Name list box, select SdShowMsg. The SdShowMsg function displays a message to your end user.

   To learn more about the SdShowMsg function, click the Help button.

9. Click the Next button. The next panel of the Function Wizard opens.
10. In the szMsg edit box, enter the text "Checking system requirements..." (including the quotation marks).
11. Click the Finish button. The function call is pasted into your script.
12. Put the following script code in the line after the `CheckRequirements();` line:

    ```
    SdShowMsg( "Checking system requirements...", FALSE );
    ```

    Use the Function Wizard, or cut and paste the previously inserted line and edit it.
13. Add the line `#include "Sddialog.h"` before the program statement and the line `#include "Sddialog.rul"` after the endprogram statement. These must be included for SdShowMsg to work.

Build your disk images

The Media Build Wizard lets you build disk image folders appropriate for any distribution medium, and it lets you specify the languages and operating systems for that build.

1. In the project workspace, click the Media pane tab.
2. Double-click the Media Build Wizard icon in the Media pane. The Media Build Wizard opens.
3. Enter "Test App" in the Media Name field of the Media Name panel (do not include the quotation marks). Click the Next button.
4. In the Type field of the Disk Type panel, select 3.5" Diskettes - 1.44 Mbytes. Click the Next button.
5. Select Full Build in the Build Type group box of the Build Type panel. Uncheck the Review Report Before Build checkbox. Click the Next button.
6. The purpose of tag files is to mark each disk of a multi-disk distribution; you can place application information in them if you wish. In the Tag File panel, do the following (do not include quotation marks):
   a) Enter "Test Company" in the Company Name field.
   b) Enter "Test App" in the Application Name field.
   c) Select "Development Tool" in the Product Category field.
   d) Leave the Misc field blank. Click the Next button.
7. In the Platforms panel, select all platforms: click the first platform in the list, then shift-click the last platform in the list. Click the Next button.
8. The Summary panel displays a summary of the choices and entries you made. Click the Finish button.
9. When the Media Build Wizard finishes building your disk images, the Finish button will come back into focus. Click Finish and InstallShield will add your new disk images (and report and log files) to the Media tree in the Media pane.

Run your new setup

1. From the Build menu, select Run Setup. Your setup will compile and run.
2. Once the setup is complete, observe how the changes you made to the setup have affected the installed application.

What do I do next?

Once you've looked over the information in "About setups" and gone through one or both tutorials, where do you go from there?

1. Start your setup project: run the Project Wizard or copy an InstallShield template.
2. For help with the various tasks involved in creating a setup, or with particular portions of the InstallShield IDE, see Getting Results. We suggest you start with the "Overview" topics to get the big picture.
3. For help with the InstallScript language as you write your script, see the Language Reference.
4. If your setup doesn't work right the first time, see the Visual Debugger Help.

For unified access to all InstallShield help, see the Help Library.

And what are the odds of that?

Migrating from InstallShield3

Doubtless you've already noticed that InstallShield5 is radically different from previous InstallShield versions. For the most part, forget what you knew.

All the work of creating a setup is now done in one single location—the InstallShield IDE. For example, no longer do you need to create, compress, and split compressed library files using DOS-based programs—InstallShield5 wizards take your data files and create your file media library for you.

Fewer setup tasks require scripting in InstallShield5. Gone are "file sets" and all the work it took to display a progress indicator during file transfer.

If you're unfamiliar with InstallShield5, the first step in learning the product is to run through the tutorials in this help file (Getting Started). You might decide that you want to recreate your setup in InstallShield5 by using the Project Wizard or by using an InstallShield5 project template. If so, you can discard the setup you created with InstallShield3, and continue creating your setup as a new project. (InstallShield5 cannot use the InstallShield3 Data.z compressed library files or your _setup.lib file, and you'll already have a new setup script.)

If you want to incorporate your existing InstallShield3 setup into your new InstallShield5 project, follow the steps below as they apply to your setup:

You can discard everything from your InstallShield3 setup except for your setup script and any resources, such as bitmaps and DLLs, that you might want to reuse. InstallShield5 cannot use the Data.z compressed library files, tag files, packaging list files, or _setup.lib from an InstallShield3 setup.

1. In the InstallShield IDE, select File | New.
2. In the General Tab, highlight Blank Setup. Click OK. InstallShield creates a skeleton for your new project, entitled Blank Setup-1 by default.

   After you've closed your project, you can rename it. Select Projects from the View menu. Right-click on the icon for your project and select Rename. To change other properties, such as the target platform(s) and language(s), right-click on the icon and select Properties.
3. Assign files to file groups.
4. Assign file groups to components.
5. Assign components to setup types. In the project workspace, open the Setup Types pane and highlight the desired component. Select or deselect the desired components in the Setup Type window on the right.
6. Include Setup.bmp (and Setup16.bmp) in your new setup.
7. Include other resource files in your setup by placing them in the Setup Files pane in the folders for the appropriate target language(s) and platform(s). At run time, you can still access these files with the SUPPORTDIR path.
8. Import your setup script, and any included files, into the new project.

   Delete the existing Setup.rul.

   a) In the Scripts pane, right click the Setup.rul icon. A popup menu appears.

   b) From the popup menu, select Delete.

   Now import the files.
9. Update your InstallShield3 setup script. There are a number of significant changes between an InstallShield3 setup script and an InstallShield5 setup script.
10. Compile your script by selecting Compile from the Build menu. Debug, if necessary, by selecting Debug Setup from the Build menu.
11. Build your setup using the Media Build Wizard. Start the wizard by selecting Media Build Wizard from the Build menu.
12. Test your setup. Select Run Setup from the Build menu.
13. Copy your setup to the distribution media by selecting Send Media To from the Build menu. Test.

InstallShield
www.installshield.com

Updating an InstallShield3 setup script

Since you're already familiar with InstallScript, perhaps the best way for you to get a feel for the changes in InstallShield5 is to look through a new setup script. Open one of the template projects, and click Setup.rul in the Scripts pane.

Right away you'll notice that the declare keyword and start label are no longer necessary. Maybe you're wondering where most of the #define statements went. And you won't see a single call to FileSetPerform. A lot has changed.

InstallScript enhancements

InstallShield offers several new features that should make the process of writing scripts even easier:

- New InstallScript operators and data types similar to those you already know from C language programming.
- A new, more robust InstallScript compiler with more specific and meaningful error and warning messages.
- Added functions and options.

{button ,JI(`Getstart.HLP>(w95sec)',`New_functions_in_InstallShield5')} New functions in InstallShield5

{button ,JI(`Getstart.HLP>(w95sec)',`Functions_changed_in_InstallShield5')} Functions changed in InstallShield5

{button ,JI(`Getstart.HLP>(w95sec)',`Obsolete_functions_removed_from_InstallShield')} Obsolete functions removed from InstallShield

String identifiers

String identifiers are those @IDENT_NAME things you see in the script where you'd expect to find constants. At run time InstallShield replaces the identifiers with the values assigned in the string table (in the IDE's Resources pane).

Among the advantages of defining string constants in string tables rather than #define statements is that you can create a different string table for each target language. When you build your setup with the Media Build Wizard, InstallShield includes only the correct string table(s) for the desired target language(s).

{button ,JI(`Getstart.HLP>procedur',`Replace_define_statements_with_string_table_entries')} Replace string constant #define statements with string table entries

File transfer

File sets—and all the Compress... and FileSet... functions—are happily gone. File transfer now begins when you call the ComponentMoveData function.

Information about your file groups, components, and setup types is stored within the file media library that you create in the InstallShield IDE.

The component and setup type dialogs are designed to display information about your file media library and return information that ComponentMoveData can automatically process.

The bottom line is that most setups need only call three functions to get setup type and component information and transfer files based on end-user selections. For more information, see The minimum function calls necessary to install component selections.

You can create script-generated components at run time—that is, create component choices not defined in the original file media library. Handling script-generated components is more complicated, involving numerous calls to the component functions to simulate IDE selections. Be aware that, although many of the names are still the same, the component functions work very differently in InstallShield5. In fact, many of them take completely different parameters.

Replace string constant #define statements with string table entries

First put the string constants and values in a string table.

1. If the string constant #define statements are in a file that is not displayed in the project workspace's Scripts pane, insert it into the pane.
   a) Right-click in the Scripts pane. A popup menu appears.
   b) Select Insert Files... from the popup menu. The "Insert Files into Script Files" dialog box opens.
   c) Select the file in the "Insert Files into Script Files" dialog box.
   d) Click the OK button.
2. In the Scripts pane, double-click the file containing the string constant #define statements. The file opens in a script editor window.
3. If the Resources - String Table\<language> window is not open, open it by doing the following:
   a) Click the tab of the Resources pane.
   b) Double-click the desired language icon under the String Table icon.

For each #define statement defining a string constant, do the following:

4. Copy the string constant name to the string table.
   a) In the script editor window, select the string constant name (for example, MY_STRING_CONSTANT).
   b) Press Ctrl+C to copy the selected string constant name.
   c) Right-click in the Resources - String Table\<language> window. A popup menu appears.
   d) Select New in the popup menu. The Add/Modify String Entry dialog box opens.
   e) In the Add/Modify String Entry dialog box, press Ctrl+V to paste the string constant name into the String Identifier edit box.
   f) Click the OK button.
5. Copy the string text to the string table.
   a) In the script editor window, select the string text (for example, My String Text). Do not include the quotation marks surrounding the string text.
   b) Press Ctrl+C to copy the selected string text.
   c) In the Resources - String Table\<language> window, double-click the string constant name you previously added. The Add/Modify String Entry dialog box opens.
   d) Place the cursor in the Value edit box.
   e) Press Ctrl+V to paste the string text into the Value edit box.
   f) Click the OK button.
6. In your script, change all occurrences of the string constant to the correct format: @<identifier>.
   a) With the script editor window active, press Ctrl+H. The Replace dialog box opens.
   b) In the "Find what" edit box, enter the string identifier (for example, MY_STRING_CONSTANT).
   c) In the "Replace with" edit box, enter the string identifier with an "at" symbol (@) prepended (for example, @MY_STRING_CONSTANT).
   d) In the "Replace in" area, make sure the "Whole file" radio button is selected.
   e) Click the Replace All button.

Repeat steps 4, 5, and 6 for each string constant.

New functions in InstallShield5

ComponentCompareSizeRequired
Determines if enough disk space exists for all components.

ComponentError
Returns additional error information when a Component function fails.

ComponentFileEnum
Enumerates the files associated with a component.

ComponentFileInfo
Retrieves all available information about a file associated with a specified component.

ComponentFilterLanguage
Enables and disables filtering based on language.

ComponentFilterOS
Enables and disables filtering based on operating system (OS).

ComponentGetData
Retrieves all available information from the specified file media library, script-generated component set, or component.

ComponentMoveData
Transfers and decompresses files of selected components in the specified file media library.

ComponentSetData
Sets the field data of the specified component.

ComponentSetTarget
Specifies a user-defined variable to place in a component's <TARGETDIR> field.

ComponentSetupTypeEnum
Enumerates all setup types associated with the specified file media library or script-generated component set.

ComponentSetupTypeGetData
Retrieves data associated with a specified setup type that has been created in the InstallShield IDE.

ComponentSetupTypeSet
Selects all components associated with the specified setup type.

ComponentValidate
Validates the password of the entire file media library or a specified component.

DeinstallSetReference
Specifies a reference file to be checked before the uninstallation process begins.

PlayMMedia
Plays a sound or AVI file.

SdSetupTypeEx
Displays a dialog box that allows you to select the type of installation to perform.

SetDisplayEffect
Sets the display effect for bitmap or metafile images.

Functions changed in InstallShield5

All the functions discussed below existed in previous versions of InstallShield and still exist in InstallShield5, but their syntax has changed.

The following four functions used to take as their first argument the name of a list containing component names. In InstallShield5, they take as their first argument the name of a file media library or script-generated component set.

ComponentAddItem
ComponentGetItemSize
ComponentIsItemSelected
ComponentSelectItem

The ComponentTotalSize function used to take two arguments (the first of which was the name of a list containing component names). In InstallShield5, it takes four arguments (the first of which is the name of a file media library or script-generated component set).

Obsolete functions removed from InstallShield

New InstallShield functionality makes the following functions obsolete, so they have been removed from InstallShield. Click the button preceding a function name to learn how to replace that function in your old script.

{button ,JI(`Getstart.HLP>(w95sec)',`Replace_ComponentGetItemInfo')} ComponentGetItemInfo

{button ,JI(`Getstart.HLP>(w95sec)',`Replace_ComponentSetItemInfo')} ComponentSetItemInfo

{button ,JI(`Getstart.HLP>(w95sec)',`Replace_CompressAdd')} CompressAdd

{button ,JI(`Getstart.HLP>(w95sec)',`Replace_CompressDel')} CompressDel

{button ,JI(`Getstart.HLP>(w95sec)',`Replace_CompressEnum')} CompressEnum

{button ,JI(`Getstart.HLP>(w95sec)',`Replace_CompressGet')} CompressGet

{button ,JI(`Getstart.HLP>(w95sec)',`Replace_CompressInfo')} CompressInfo

{button ,JI(`Getstart.HLP>(w95sec)',`File_sets_are_obsolete')} FileSetBeginDefine

{button ,JI(`Getstart.HLP>(w95sec)',`File_sets_are_obsolete')} FileSetEndDefine

{button ,JI(`Getstart.HLP>(w95sec)',`File_sets_are_obsolete')} FileSetEndPerform

{button ,JI(`Getstart.HLP>(w95sec)',`File_sets_are_obsolete')} FileSetPerform

{button ,JI(`Getstart.HLP>(w95sec)',`File_sets_are_obsolete')} FileSetPerformEz

{button ,JI(`Getstart.HLP>(w95sec)',`Replace_FileSetReset')} FileSetReset

{button ,JI(`Getstart.HLP>(w95sec)',`File_sets_are_obsolete')} FileSetRoot

{button ,JI(`Getstart.HLP>(w95sec)',`Replace_Split')} Split

Replace ComponentGetItemInfo

Replace ComponentGetItemInfo(szCompList, szComponent, nInfo, svData) with ComponentGetData(szMedia, szComponent, nInfo, nvData, svData). Specify the ComponentGetData arguments as follows:

szMedia
Replaces szCompList (the name of a list containing component names). Specifies the name of the file media library or script-generated component set; use a string, a string identifier, or the system variable MEDIA.

szComponent
Same as in ComponentGetItemInfo. Specifies a component; to specify a subcomponent, use path name syntax, for example, "Component\\Subcomponent".

nInfo
As in ComponentGetItemInfo, specifies the kind of data retrieved. Available values have changed.

nvData
As in ComponentGetItemInfo, for certain values of nInfo InstallShield writes a numeric value to this variable.

svData
As in ComponentGetItemInfo, for certain values of nInfo InstallShield writes a string value to this variable.

Replace ComponentSetItemInfo

Replace ComponentSetItemInfo(szCompList, szComponent, nInfo, szData) with ComponentSetData(szMedia, szComponent, nInfo, nData, szData). Specify the ComponentGetData arguments as follows:

szMedia
: Replaces szCompList (the name of a list containing component names). Specifies the name of the file media library or script-generated component set; use a string, a string identifier, or the system variable MEDIA.

szComponent
: Same as in ComponentSetItemInfo. Specifies a component; to specify a subcomponent, use path name syntax, for example, "Component\\Subcomponent".

nInfo
: As in ComponentSetItemInfo, specifies the kind of data to be set. Available values have changed.

nData
: As in ComponentSetItemInfo, to set numeric data, enter the value in this parameter; set szData equal to the null string ("").

szData
: As in ComponentSetItemInfo, to set string data, enter the value in this parameter; set nData equal to 0 (zero).

Replace CompressAdd

Delete calls to CompressAdd from your script. Add files (or file groups or components) in the IDE before building the compressed cabinet file.

{button ,AL(`Associate file groups with a component GS;Create a new component GS;Organize my files into file groups GS',0,`',`')} See also

Replace CompressDel

Delete calls to CompressDel from your script. Delete files (or file groups or components) in the IDE before building the compressed cabinet file.

{button ,AL(`Delete a component GS;Delete a file from a file group GS;Delete a file group GS;Disassociate a file group from a component GS',0,`',`')}See also

Replace CompressEnum

Replace CompressEnum(szLibrary, szQuery, listFiles, nOption) with ComponentFileEnum(szMedia, szComponent, szQuery, listFiles, nOption). Specify the ComponentFileEnum arguments as follows:

szMedia
Replaces szLibrary (the name of a compressed library file). Specifies the name of the file media library or script-generated component set; use a string, a string identifier, or the system variable MEDIA.

szComponent
Specifies the component in which to look for the file(s) specified by szQuery. To search in all components, set szComponent equal to the null string ("").

szQuery
Specifies the files and file groups to be searched. In CompressEnum, the szQuery string was interpreted as follows:

"File"	File.
"Path\\File"	Path in the compressed library, and file.

In ComponentFileEnum, the szQuery string is interpreted as follows:

"File"	File group. (Searches for all files in the file group.)
"File Group\\File"	File in a file group.
"File Group\\Path\\File"	Path and file in a file group.

You can use wild cards; for example, "MyGroup\\*.txt" searches for all .txt files in the file group MyGroup, and "*\\*.exe" searches for all .exe files in all file groups.

listFiles
As in CompressEnum, the name of a list to which the names of the found files are written.

nOption
As in CompressEnum, specifies whether or not to include subfolders in the search. Set equal to INCLUDE_SUBDIR to search subfolders or NO_SUBDIR to not search subfolders.

Replace CompressGet

Replace CompressGet(szLibrary, szFiles, nOption) with ComponentMoveData(szMedia, nvDisk, 0). Specify the ComponentMoveData arguments as follows:

szMedia
Replaces szLibrary (the name of a compressed library file). Specifies the name of the file media library or script-generated component set; use a string, a string identifier, or the system variable MEDIA.

nvDisk
The number of the next required distribution disk is written to this variable. (InstallShield handles multiple-disk setups automatically; this variable may be useful for troubleshooting.)

The szFiles parameter is no longer necessary. ComponentMoveData automatically transfers all components that have been selected, either by your end user in a component dialog or by you with script calls to ComponentSelectItem or ComponentSetupTypeSet.

{button ,AL(`ComponentSelectItem GS;ComponentSetupTypeSet GS',0,`',`')} See also

Replace CompressInfo

Replace CompressInfo(szLibrary, szFile, nInfo, nvData, svData) with ComponentFileInfo(szMedia, szComponent, szFile, nInfo, nvData, svData). Specify the ComponentFileInfo arguments as follows:

szMedia
: Replaces szLibrary (the name of a compressed library file). Specifies the name of the file media library or script-generated component set; use a string, a string identifier, or the system variable MEDIA.

szComponent
: Specifies the component in which to look for the file or file group specified by szFile. To search in all components, set szComponent equal to the null string (`""`).

szFile
: Can, as in CompressInfo, specify a file about which you want information. Can also specify a file group.

nInfo
: As in CompressInfo, specifies the kind of data retrieved. Available values have changed.

nvData
: As in CompressInfo, for certain values of nInfo InstallShield writes a numeric value to this variable.

svData
: As in CompressInfo, for certain values of nInfo InstallShield writes a string value to this variable.

File sets are obsolete

File sets are no longer necessary, and file set functions are no longer supported. Replace script code of the following sort:

```
FileSetBeginDefine( "MyFileSet" );
   CompressGet( "Data.z", "*.exe", COMP_NORMAL );
   FileSetRoot( "MyFileSet", "A:\\MyDLLs\\" );
   CompressGet( "Data.z", "*.dll", COMP_NORMAL );
FileSetEndDefine( "MyFileSet" );

FileSetPerformEz( "MyFileSet", 0 );
FileSetEndPerform( "MyFileSet" );
```

with code like the following:

```
ComponentSelectItem( MEDIA, "MyDLLs", TRUE );
ComponentMoveData( MEDIA, nvDisk, 0 );
```

{button ,AL(`ComponentMoveData GS;ComponentSelectItem GS;Replace CompressGet',0,`',`')}   See also

Replace FileSetReset

Since you'll no longer use file sets, you'll no longer need to reset them. Delete the call to FileSetReset and replace your file set.

Replace Split

Delete calls to Split from your script. Set your distribution disk size in the Media Build Wizard's Disk Type panel. Leave blank space on a particular disk with the Advanced build Type property sheet's Disk Information page.

{button ,AL(`Leave extra space on one of my disks GS;Overview: Building Distribution Media',0,`',`')} See also

What's new

InstallShield5 has a radically new look—an integrated development environment in which you do many setup creation tasks by pointing and clicking.

{button ,JI(`GETSTART.HLP>whatnews',`Starting_a_new_setup_project')} Starting a new setup project

{button ,JI(`GETSTART.HLP>whatnews',`Organizing_your_file_transfer')} Organizing your file transfer

{button ,JI(`GETSTART.HLP>whatnews',`Building_your_distribution_media')} Building your distribution media

Script writing has been made much easier in InstallShield5 with new tools.

{button ,JI(`GETSTART.HLP>whatnews',`Function_Wizard')} Function Wizard

{button ,JI(`GETSTART.HLP>whatnews',`Script_editor')} Script editor

In addition to new tools for setup creation, InstallShield5 provides new InstallScript functionality.

{button ,JI(`GETSTART.HLP>whatnews',`Multimedia_and_256_color_bitmaps')} Multimedia, 256-color bitmaps, and special effects

{button ,JI(`GETSTART.HLP>whatnews',`Remote_registry_connections')} Remote registry connections

{button ,JI(`GETSTART.HLP>whatnews',`Operators_data_structures_and_arrays')} Operators, data structures, and arrays

{button ,JI(`Getstart.HLP>whatnews',`String_conditionals_in_switch_statements')} String conditionals in switch statements

Starting a new setup project

As a first big step in creating your setup, launch the Project Wizard. In seven simple point-and-click steps you can create a fully functional InstallShield setup for your application.

Organizing your file transfer

By pointing and clicking, you can organize your application's files into file groups—and arrange those file groups as components and setup types to give your end users exactly what they need.

Building your distribution media

To create disk images for your distribution media, launch the Media Build Wizard. In six simple point-and-click steps you can create disk images ready for copying to floppy disks or CD-ROM.

Function Wizard

To call an InstallScript function in your setup script, launch the Function Wizard. Choose the function you want (with the help of brief or detailed online information), fill in its parameters with edit boxes and drop-down lists, and click the Finish button to paste the complete function call into your script.

Script editor

The new InstallShield script editor offers syntax coloring of functions, keywords, constants, and other InstallScript syntax elements. It also provides find and replace dialog boxes, bookmarks, and line markers to assist you in editing your script. And you can get online help for any InstallScript function by placing your cursor in the function and pressing F1.

Multimedia, 256-color bitmaps, and special effects

InstallShield5 lets you do the following during your setup:

- Play AVI movies.
- Play WAV/MIDI sound files.
- Display 256-color bitmaps.
- Display 256-color gradient backgrounds with built-in or custom colors.
- Display bitmaps with special effects such as fading and filling.

Remote registry connections

For a setup run across a network, InstallShield5 can create a connection to a remote registry.

Operators, data structures, and arrays

InstallScript now includes the following:

- Address (&), indirection (*), member (.), and structure pointer (->) operators.
- Nested data structures.
- Character array indexing.

String conditionals in switch statements

Switch statements can now evaluate string expressions; for example:

```
switch (svSetupType)
   case "Custom":
      MessageBox( "Custom", INFORMATION );
   case "Compact":
      MessageBox( "Compact", INFORMATION );
   default:
      MessageBox( "Typical", INFORMATION );
endswitch;
```

What's what in the InstallShield IDE

Click here {button ,JI(`GETSTART.HLP>screen',`InstallShield_IDE_initial_view')} for a graphical guide to the InstallShield IDE as it initially appears when you launch InstallShield.

There's even more to see. To continue, do the following:

1. On the toolbar, click the New button . The New dialog box opens.
2. In the New dialog box, click the Template tab.
3. Click the OK button to open a copy of Template One as a new setup project (named "Template One-1").

Click here {button ,JI(`GETSTART.HLP>screen',`InstallShield_IDE_project_open')} for a graphical guide to the InstallShield IDE as it now appears.

Now that you've come this far, you've completed the first few steps of the tutorial "Use an InstallShield template." Click here {button ,JI(`GETSTART.HLP',`Use_an_InstallShield_template')} to do the whole tutorial.

InstallShield IDE: initial view

This is the Projects window. It contains the Project Wizard icon, and icons for any already-existing setup projects.
Double-clicking one of the project icons opens the project workspace for that project.

This is the menu bar.

This is the toolbar.

This is the output window. It displays messages generated during script compilation and file media library building.

This is the status bar.

InstallShield IDE: project open

This is the project workspace.

This area of the IDE displays various windows in which you can view and modify the content or properties of individual parts of the open setup project, such as file groups, components, and scripts. The script editor window is shown here. To see other windows, select Tile from the Window menu in the IDE.

{button ,AL(`Component Properties window GS;File Group Properties window GS;Files window GS;Media window GS;Resources window GS;Script Editor window GS;Setup Files window GS;Setup Type window GS;Strings window GS',0,`',`')} See also

Technical support

InstallShield Corporation is committed to providing you with the best possible product support for InstallShield and all of its products. In order to get the fastest, most complete, and most accurate support, carry out the follow procedures.

Register your software

Consult technical support resources

InstallShield
www.installshield.com

Register your software

In order to receive 30 days free technical support and be eligible for our Developer Support Programs, be sure to register your software. Registering also enables you to receive the latest InstallShield technical information and notification about product releases and upgrades.

There are three ways to register your InstallShield software:

- Fill out the registration form on InstallShield's registration Web page.
- Complete the registration form presented to you during the InstallShield setup.
- Fill out and send in the registration card provided for you in the InstallShield box.

Consult technical support resources

Online help

You should first consult the online help if you have questions about InstallShield. The help files contain useful information about how to use InstallShield and explanations of its components. Take advantage of the help files' search and index features.

InstallShield Web site

Consult the InstallShield Web site for immediate access to a wealth of useful information. The site is a comprehensive resource with a range of free support options such as:

- Technical information and product news
- Sample installations
- Frequently Asked Questions
- White papers
- Maintenance releases and updates for registered software
- Information about consulting, support and professional services

This information is available to you 24 hours a day, 7 days a week.

InstallShield Knowledge Base

The InstallShield Knowledge Base is at the Web site. It contains answers to many commonly asked questions and includes new information about InstallShield that may not appear in the documentation. You can use the search engine provided on the Knowledge Base to search articles by phrases, numbers, platforms, and version. This information is available to you 24 hours a day, 7 days a week.

InstallShield newsgroups

The newsgroups are another excellent resource when you have questions about InstallShield. The newsgroups are a place where developers share tips and ideas and help each other get the most out of InstallShield. Visit the InstallShield newsgroups at news.installshield.com.

30 days free technical support

InstallShield
www.installshield.com

30 days free technical support

Upon registering your product, InstallShield provides you with free technical support for 30 days from the date of the response to your first support request. During these 30 days you will have access to our Developer Support Engineers via World Wide Web, email, fax, or phone(Monday through Friday from 9:30 AM to 5:30 PM CST):

WWW:	Support Request form
Email:	IS5Psupport@installshield.com
Fax:	(847) 240-9138
Phone:	(207) 622-6273

(Support is limited to installation assistance and answers to general InstallShield usage questions. Responses to questions relating to operating system functionality, development environments, and the operation of the various Windows technologies are not included.)

After your free support has expired, be sure to visit the Developer Support Programs Web page for information about our extended support options.