# QSetup
## Installation Suite
Manual for version 12.0

# Table of Contents

# General

QSetup from "Pantaray Research LTD.", is today's most effective and powerful setup program.
QSetup is designed to let you create high quality sophisticated installation delivery with minimum effort and no script programming.
QSetup features user-friendly and intuitive interface, that will help you create solid & dependable installations in a very short period of time.
QSetup will produce a single Self Extract installation file that can be easily downloaded from the Internet, distributed on a CD or placed on a central File Server.
QSetup can also create Split Setup, where you supply your customers with a small Setup Kernel that will download the rest of the setup file from the Internet.

QSetup is **unique** in its ability to create setup that will **Auto Update** from the Internet, when a new version of the program is available.

## Components

The program includes 3 major components:

**Composer.exe**    The compiling program that creates the installation delivery file.

**Engine.exe**    The program that will run on your customer's computer to perform the actual Installation & UnInstallation.

**Stub.exe**    The Self Extract kernel.

You will only use directly the Composer program. The other components will be added automatically to the Installation delivery by the Composer.

# Composer.exe

The Composer is the program that creates the installation delivery file - also **known** as Media file or Setup file.

Page Buttons



Perform Buttons

Help Tools...

MSI \ EXE Selection
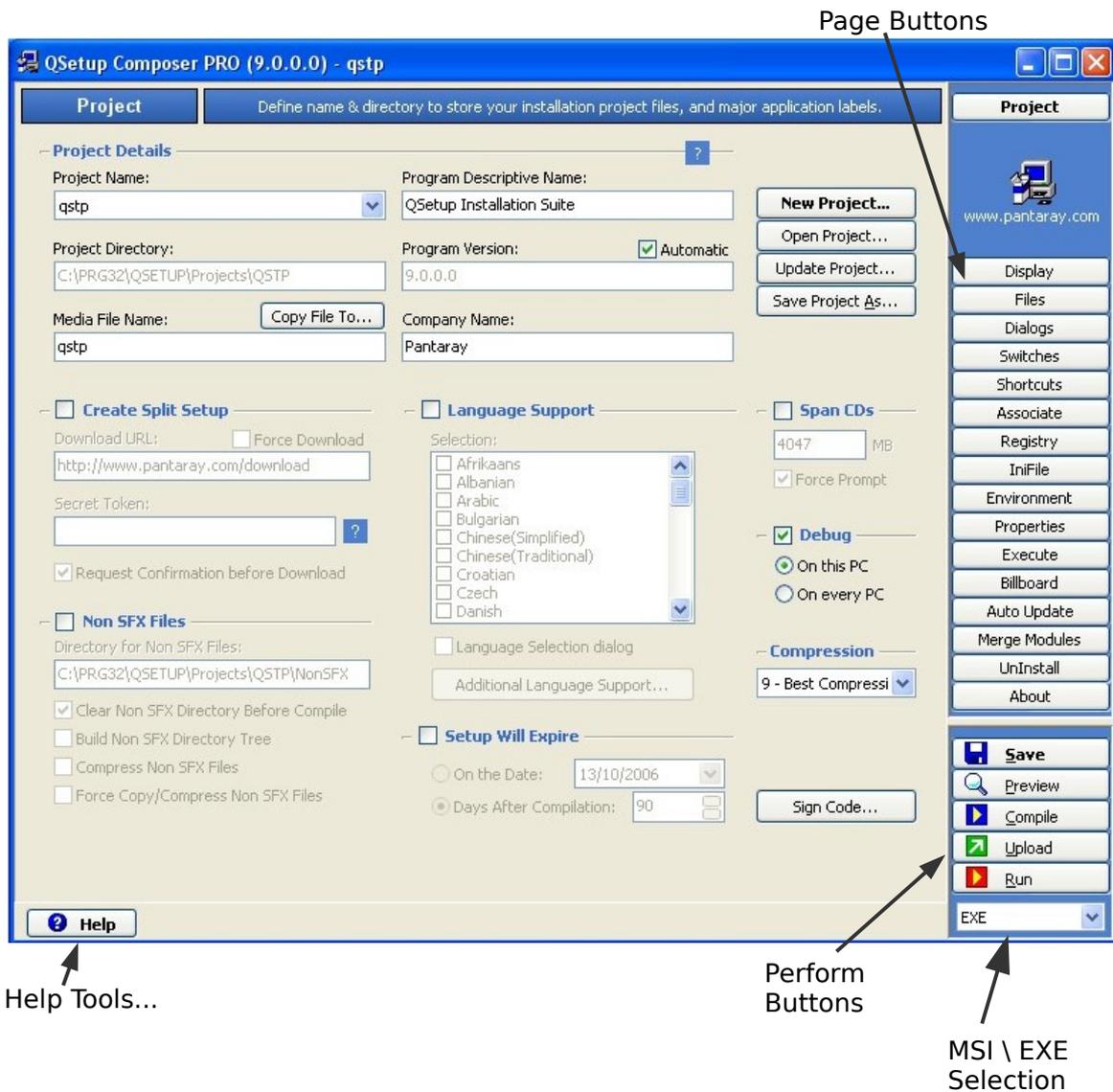
### Page Buttons

On the Top/Right side of the Composer screen you will find the following buttons: Project, Files, Display, Dialogs, Switches, Shortcuts, Associate, Registry, IniFile, Environment, Properties, Execute, Billboard, Auto Update, Merge Modules, UnInstall and About.

When ever you click one of these buttons, the Top/Left area of the composer screen will change its content to reflect the selection you made.
When you start a new Installation project go through all the pages in the order of the buttons and fill the required items in every page.

**Perform Buttons**

On the Bottom/Right side of the Composer screen you will find the following buttons: Save, Preview, Compile, Upload and Run.

**Save** - Whenever you change any data this button will become active, press this button to save the data.

**Preview** - Press this button to see what your customers will see during the installation. This is just a visual test, no actual installation will be performed on your computer.

**Compile** - When you press this button the Composer will create the actual distribution file (Media File) you will send to your customers.

**Sign Code** - After you compile your setup and before you upload it to the web, click the [Sign Code] button to open the "Sign Code" dialog and sign your setup file.

**Upload** - Use this button to Upload your setup delivery to the Internet using FTP protocol.

**Run** - When you press this button the Composer will run the actual distribution file. If required the Composer will compile the file before running it. When you click this button actual installation will be performed on your own computer.

IMPORTANT - The "Sign Code" dialog is available only in the PRO Version.

**MSI/EXE Selection**

On the Bottom/Right corner of the Composer screen you will find a special selection box. Use this selection box to select MSI setup file or traditional EXE setup file as your target setup compilation.

# Command Line Parameters

**When running the Composer**

When running the Composer you can add the following parameters on the command line:

- File Name - The full path of a QSP file name - this file will be loaded for editing or compiling.
- /Compile - Compile the current setup.
- /CompileAU - Compile the current Auto-Update setup.
- /Exit - Exit the Composer (will be performed after Loading and/or compiling).
- /EXE - Compile an EXE type setup.
- /MSI - Compile a MSI type setup.
- /PATH_1=? - Force Path Alias "<PATH_1> to a desired value. (See help on Files page).
- /PATH_2=? - Force Path Alias "<PATH_2> to a desired value. (See help on Files page).
- /PATH_3=? - Force Path Alias "<PATH_3> to a desired value. (See help on Files page).

IMPORTANT - if /Exit is specified and an error has occurred during compilation, the composer will return with an ExitCode of One (1) otherwise the ExitCode will be Zero (0).

You can test the ExitCode of the composer using a batch file similar to the following:

```
@echo off
```

```
composer.exe "C:\MyDir\MyFile.qsp" /compile /exit

if ERRORLEVEL 1 goto _ERROR
if ERRORLEVEL 0 goto _NoError

:_ERROR
echo === ERROR ===
goto END

:_NoError
echo === No Error ===
goto END

:END
pause
```

### When running the setup
When running the Setup you can add the following parameters on the command line:
- /Silent - The Setup program will run without intervention of the user.
- /Hide - The Setup program will run completely in the background, presenting no dialog during the process.
- /InstallDir="Destination Directory" - Define the default Destination directory for the setup.

IMPORTANT - **/silent** & **/hide** will have no effect if the "User Information" dialog is selected.

### Variables On the Command Line
When running the Setup you can define variables on the command line.
The syntax is similar to the following:
- /[Var1]=1234 /[MyName]=John

There is no limit to the number of variables you can define in this way.
The variables you define will later be used by the "Execute Engine".

# Help

On the Bottom/Left side of the composer screen, you will find the [Help] button.
Click this button to open a small Help area on the Bottom/Left side of the composer screen.
If you click this button once again the help area will close.
You can change the size of the help area by dragging the bar on top of it.
The content of the help area will change when you select different pages.

### Help Tools
Click the **Help Tools** panel to open the "Help Tools Menu".
You can also RightClick the help area to get the same menu.

From the "Help Tools Menu" you can perform the following operations:

- **Print** - Send the current help document to the printer.
- **Find** - Search for a word or phrase in the help system. You can perform a search on the current help document or ALL the documents.

- **Show Help** - By selecting this option you will open the current help document in a separate window. The content of this help window will NOT change as you select different pages.

- QSetup Manual (pdf) - All QSetup help files are now available for download as a single PDF file.When you select this option, QSetup will first open the PDF from the internet. At the same time it will also download it to your disk. Next time you attempt to open the PDF QSetup will open it from the local storage, thus you can read this PDF also when you are offline.

# Project

A project is the collection of data from which the target Setup program is produced by the Composer program.

To start a new project - click the **[New Project...]** button.

## Project Definition

**Project name**
Give your project a simple name that will help you identify the current project. Examples can be "Word2000" or "Excel97".

**Project Directory**
The project directory is the place where the output of the Composer program will be placed. By default the Composer program will create a subdirectory with the name of the project starting from the "Projects" subdirectory under the Composer program directory.

**Media File Name**
This will be the name of the target Setup output file (the installation file you will deliver to your customers). This file is created by the composer when you press the **[Compile]** button (located at the Right/Bottom of the composer screen). This file will be placed in the "Project Directory".

**Copy File To...**
After you compiled a Media file you can copy it to another location by clicking the **[Copy File To...]** button. You can perform the same operation by Right Clicking the **[Compile]** button and selecting: "Copy Media To...".

**Program Descriptive Name**
This is a descriptive name that is normally composed of 2 or 3 words (some times refers to as the "Product Name").
This name will be used by the Setup program when creating shortcuts to your program in the Start/Program menu, and other places.
**Example -** "Word for Windows" or "Netscape Navigator".

**Program Version**
Enter here the version of your program. If you check the **Automatic** check box, the composer program will constantly update this field by reading the version stamp from the program's target executable resource table (defined in the Files page). This field is used during installation to create some registry keys.

**Company Name**
Enter here the name of your company. This field is used during installation to create some registry keys.

VERY IMPORTANT
You must fill all the above mentioned data items (6). QSetup is using this information when constructing registry entries. If you omit some or all of the items, your program might not install properly under Windows.

**Project Handling Buttons**

**New Project...**
Use this button to create a new Setup project.

**Open Project...**
Use this button to open an existing project for editing. Generally it will be enough to select the existing project from the drop down list of Project Names.

**Update Project...**
Use this button if you want to modify the "Project Name" and/or the "Project directory".

**Save Project As...**
Use this button to create a duplicate of the current project.

## Create Split Setup

QSetup allows you to create 2 types of setup delivery:

1. **Regular Setup** - only one Self Extract *.EXE file (up to 4GB size).

2. **Split Setup** - one small KERNEL file (~170KB) and the rest of the data in another SPLIT file (*.SPLIT) - no size limit.

The idea behind Split Setup is that the user run the KERNEL file, and this file downloads the rest of the data from the Internet, or from a central file server, and then proceed with the setup process.

**Download URL**
Enter here the URL of your website. When run, the KERNEL will download the SPLIT file from the website using HTTP protocol.
If your SPLIT file is stored in a subdirectory of the website add the name of the subdirectory to the URL.

Example: **http://www.microsoft.com/download.**
You can also specify here a directory path. This option is useful if you want to store the SPLIT file in a central file server.

**Force Download**
Usually when the KERNEL starts it will look for the SPLIT file in the same directory where it was started from. If the file is found the KERNEL will use it, if not the KERNEL will look for the file in the specified "Download-URL".
If you want to instruct the KERNEL to always use the "Download-URL" check the "Force Download" CheckBox.

**Secret Token**
The secret token is a means for added security. It is encrypted and embedded in both the KERNEL file and the SPLIT file. Setup will only take place if the "Secret Token" of both files matches.
Enter here any text you want.

IMPORTANT - for added security, the Secret Token is stored in your registry - not the QSP file.

**Request Confirmation Before Download**
Instruct QSetup to display a message box requesting confirmation before downloading the SPLIT file.

**IIS Server**
If you are hosting your website on an IIS server, you will have to set some new MIME types for "Split Setup" to work properly. For more information go to the following page: www.pantaray.com/iis.html.

# Non SFX Files

Usually QSetup creates a single Self-Extract (SFX) file. This type of setup delivery is best suited for small setup files (several MB) that are usually downloaded from the Internet. However if you attempt to create a large setup delivery (tens or hundreds of MB), you would probably deliver your setup on a CD.
For performance reasons, when creating a large CD Setup we recommend that you create a small SFX file that include only small files and leave the large files (Usually Images & Movies) on the CD.

To create such a setup, you only need to mark the large files - on the "Files" page - as "Excluded From SFX File".
All the files marked this way will not be included in the SFX file.
Later when you burn your CD you will need to include those files on the CD as well.
The files must be located on the CD in the same directory where the SFX file is located, or in any subdirectory of this directory.

IMPORTANT - You must make sure that all the "Non SFX Files" have unique names (unless you check the "Build Non SFX Directory Tree" option - described later).

**Directory for Non SFX Files**
If you define this directory then during the Compile process, all the Non-SFX files will be copied to this directory.
By default this directory will receive the name "NonSFX" and will be located inside your "Project Directory".

**Clear Non SFX Directory Before Compile**
If this option is checked, all the files (and directories) located in the "NonSFX" directory will be erased before compilation.

**Build Non SFX Directory Tree**
If this option is checked, then - during compilation - QSetup will attempt to build inside the "NonSFX" directory a new directory tree that will look like the directory tree your setup will build on the target PC during setup. Also during compilation QSetup will place every Non SFX file inside its designated directory. Later on when you burn your CD all that you have to do is copy the CONTENT of the "NonSFX" directory (Files & SubDirectories) to the CD.

IMPORTANT - if your setup includes several files with the same name, the only way to deliver them as NonSFX is by checking this option.

### Compress Non SFX Files

When this option is checked, all the NonSFX files will be compressed during compilation. If you select this option, you must NOT allow your customer to use the "Partial" setup option (Described on the "Dialogs" page).
All compressed files will have the extension "._z" added to their name.

### PLEASE NOTE

Compressing the NonSFX files will reduce the total size of files on the CD, however this might complicate the setup process as files must be UnCompressed during installation and then copied to the Target Directory.

### Force Copy/Compress Non SFX Files

QSetup will copy and/or compress Non SFX Files to the Non SFX Directory only if a file modified since the last time it was copied. If you want to Force the copy/compress operation Check this CheckBox. QSetup will compare files based on their time stamp.

## Language Support

QSetup includes a comprehensive infrastructure for Multilingual support. Currently more then 30 languages are supported with more to come.

**IMPORTANT** - Language Support is NOT available in the LITE Version.

### Selection

Check all the languages you want your project to support.
During Installation the Setup program will select the required language based on the **LOCALE** information found in the operating system. If a match is not found, the Setup program will default to English.

If you check **ONLY ONE** language this one will be the selected language - no check performed.

### Adding more languages

The language support is based on a text file in the form of an INI file. The files are located in the subdirectory LANG.
If you want to create your own LANGUAGE file, you must copy the "English.lng" file to a new file with a name like "French.lng" and then edit the new file. Detailed instructions can be found in the file "Instructions.txt" located in the LANG directory.
With similar techniques you can also easily edit an existing language file.

### Language Files on the Web

We are hosting a special page on our WebSite, were we post various language files that were contributed by users of the program, to visit this page go to www.pantaray.com/language.html.

### Language Selection Dialog

By default, During Installation, QSetup will select the required language based on the **LOCALE** information found in the operating system. If you check the option "Language Selection Dialog", QSetup will present to the end user a selection dialog were he can select the required installation language. This dialog will appear at the very beginning of the installation (just after extract).

### Additional Language Support

QSetup provides additional tools for a full language support. Using this dialog you can provide the following PER LANGUAGE information:

- License Agreement Text File.
- ReadMe Text File #1
- ReadMe Text File #2
- Program Descriptive Name
- Company Name
- Top Label
- Top Label 2
- Bottom Label

Using this dialog you can provide "License Agreement" and/or "ReadMe" files in different languages as well as names and labels.
Add information only for the languages you are interested in.
If some fields are left unused QSetup will use default values from the relevant composer screen. For instance if you leave "Readme Text File #1" empty, QSetup will use the relevant information found on the "Dialogs" page.

**Testing Language Support**
To properely test your multilingual setup you must adjust your operating system to the language under test.
For more info read the following link: www.pantaray.com/language.html#testing.

# Setup Will Expire

Use this option limit the time your setup will be functional.
You have 2 options:

- Set a FIXED date for expiration.
- Set a number of days after you compile the setup.

When this option is set and the expiration date has elapsed, your user will not be able to run the SETUP PROGRAM any more.

# Span CDs

By default QSetup will produce a single Self Extract file.
Sometimes the single file might grow too big to fit on a single CD or other distribution media.
Using this option you can split the single file into several files based on size.
Check this option and set the maximum size of each file (for CDs we recommend 650 MB).

You can set here any value also a fractional number like 1.4 MB if you want to use diskettes.
Don't enter values smaller then 0.6 MB.

**Prompting for Next Disk**
When extracting files, QSetup will prompt the user for the next disk only if the disk is not found in the Origin directory. Thanks to this behavior, an IT Manager can copy ALL the CDs to one directory and instruct all users to install from this directory without prompting for the next disk.

**Force Prompt**

This CheckBox is valid only on the developers station. Using this option the developer of the setup can get a better feeling of the setup behavior even though all the setup files are located in one directory.

**Burning a CD**
Let's say your "Media File Name" is: "MySetup", and you have checked the option "Create Autorun.inf file" on the "Switches" page.

After you compile a SpanCDs setup you will have in the project directory a list of files similar to the following:
- autorun.inf
- MySetup.exe
- MySetup.exe.001
- MySetup.exe.002
- MySetup.exe.003

When burning this sample you will need 3 CDs.

On the first CD you will place the following 3 files:
- autorun.inf
- MySetup.exe
- MySetup.exe.001

On the second CD you will place the file: MySetup.exe.002.
On the third CD you will place the file: MySetup.exe.003.
- Typically your setup will include 1 or more files with the extension .exe.00?.

# Debug

The "Debug" option is an important tool that will help you when you develop your setup.
When this option is checked a special popup window will open when you run the setup, and a list that describe the steps of the setup will appear in this window.

**On this PC**
When this option is checked the "Debug" window will popup only on the PC on which the setup is being developed. This is the recommended option.

**On every PC**
When this option is checked the "Debug" window will popup only every PC.

# Compression Level

When creating the Setup file, QSetup compress the file to reduce the total size of the file.
Using this selection box you can select among several levels of compression.
The better the compression the longer it will take to compress the file.
The **Default** option provides the best compromise between speed & compression.

# Display

In this page you will define what your customer will see during the installation process. While editing this page you can constantly click the [Preview] button (located at the Right/Bottom of the composer screen) to monitor the results.

## Setup Background

**Add Setup Background**
If you check this Checkbox then a colored background (Normally with gradient) will cover the entire screen during installation.

**3D Frame**
Check this option to add a standard 3D Windows Frame around the Setup Background.

**Top Label**
During installation the Setup program will display this label at the top of the screen.
Use this label to display a descriptive name of your program.
You can control the appearance of the label by selecting: Side and the following Font attributes: Name, Size, Style, Color and Script.

> **Important** - When selecting a font make sure you select a Windows Generic font like Tahoma, Verdana, Arial, Time New Roman etc... However if the font you selected is not found on the target machine, the Setup program will default to "Arial".

**Add Version**
When this option is checked, the program version (as entered on the "Project" page) will be added to the "Top Label" in smaller size fonts.

**Top Label 2**
During installation the Setup program will display this label at the top of the screen - just under the "Top Label".

**Bottom Label**
During installation the Setup program will display this label at the bottom of the screen. Usually you will use this label to display Copyright Notice of your program.
You can control the appearance of the label by selecting: Side and the following Font attributes: Name, Size, Style, Color and Script.

Click the characters © and ® to insert them in to the label while editing.

**Design**
Select any of the following designs:
- Gradient
- Light Source (Right)
- Light Source (Left)
- Windmill (Right)
- Windmill (Left)
- XP Background

- Rectangles (Right)
- Rectangles (Left)
- Random Rectangles (Right)
- Random Rectangles (Left)

**Colors**
Use these color settings to define the colors of the background.
The function of each color settings vary according to the design selected.
If you prefer a solid color background, uncheck the second Checkbox.

**Background Bitmap File Name**
You can add a bitmap image to be displayed on the background during installation.
The image must be of a Bitmap type (*.BMP).

**Location**
Click the **[Location]** Button to define the location of the bitmap.
Select any of the following locations:
    Center, Top/Left, Top/Right, Bottom/Left, Bottom/Right,
    Top/Center, Bottom/Center, Left/Center, Right/Center,
    Entire Screen.

When you select **Entire Screen** the image will be stretched to cover the entire screen.

Use the DX & DY values to shift the image horizontally and vertically from the original location.

**Transparent**
When **Transparent** is Checked the image will be displayed as transparent. The transparent color is always the first pixel on the bitmap. Please note that on most Bitmap the first pixel is the first on the bottom line, on some bitmaps the first pixel is the first on the top line.

## Setup Dialog

**Style**
QSetup can display 2 types of Setup dialogs:
- Classic
- Modern

**Location**
Click the **[Location]** Button to define the location of the Setup Dialog.
Select any of the following locations:

Center, Top/Left, Top/Right, Bottom/Left, Bottom/Right, Top/Center, Bottom/Center, Left/Center, Right/Center.

Use the DX & DY values to shift the image horizontally and vertically from the original location.

**Image File Name**
The setup dialog includes a small default Bitmap. You can replace this bitmap by selecting a new one.
The recommended maximum size of the bitmap is:
- Classic:   Height=256, Width=120.

- Modern:   Height=307, Width=160.

The image must be of a Bitmap type (*.BMP).
You don't need to add the bitmap file to the files list, the Composer will add it automatically.

### 3D Frame
Check this checkbox to add a classic 3D Frame around the image (Valid only for Classic style).

### Image File Name Small
This selection is valid only for Modern style.
Use this option to select a small BMP file to be added to many of the dialogs in the Modern style.
The size of the image must be: Height=52, Width=52.

### Clipart
Starting from version 9.1.0.0 we provide a clipart of complimentary pairs of bitmaps.
Use this clipart to enhance the display of your setup.

### Company URL
Enter here the URL of your Company's WebSite. This URL will appear on the Bottom/Left side of the Setup Dialog. Clicking this URL will open the browser with your website.
You can direct the browser to another web page by adding a **|** (pipe sign) and the target URL.

#### Location
URL Text:   www.pantaray.com **|** http://www.pantaray.com/qsetup.html
Display:      www.pantaray.com
WebPage:  http://www.pantaray.com/qsetup.html

IMPORTANT - Company URL is available only for the PRO and STUDIO versions.


## Music File Name

QSetup has the ability to play music during the setup process.
Check this CheckBox and specify a WAVE file or a MIDI file. (*.WAV  *.MID).
The music will play during the "Copy Files" stage of the installation.

### Loops
Specify here how many times the Music file will repeat itself. You may specify **Endless** for continues paly.
Once the "Copy Files" stage is terminated, the playback will terminate also no matter how many loops you specified. (Please remember that the actual installation time will vary from PC to PC).
You may also specify here **Entire Track** in this case the setup will terminate only after the complete wave file has played to its end.

### Delay (MS)
For a small setup, the time available for the music to play might be too short. You can increase this time by adding a short delay after every file copy operation during the "Copy Files" stage.

### Slider
Check this CheckBox to display a Sliding Volume Control during installation.
The Slider has 2 optional locations.
- **Fixed** - The silder will be placed on the setup dialog.

- **Floating** - The slider can be placed anywhere on screen. Click the **[Location]** button to determine the exact location.

# Files

In this page you will define the collection of files that you will send to your customers in the Installation delivery.

## Definitions

**Application Folder**
Define here the ultimate target directory you suggest your customers as the target installation directory for your program.
The "Application Folder" consists of two parts:
- The first part is a predefined root directory like <ProgramFilesDir>.
- The second part is the target application directory inside the predefined root directory.

Normally you will enter here your company name like **Netscape**.
And optionally the name of the program in the form **Netscape\Navigator.**
Usually your customers will have the option to select another directory during the Setup process.

**Common Folder**
The "Common Folder" is usually used when your company supplies several programs that must share common files like DLLs and Database files.
The "Common Folder" consists of two parts:

- The first part is a predefined root directory, normally <CommonFilesDir>.

- In the second part you will normally enter your company name.

**Please Note** - Your customers will **NOT** have the option to select a different "Common Folder" directory during the Setup process.

**Auxiliary Folder**
The "Auxiliary Folder" is usually used when you want to place files in some other "Well Known" location like the FONTS directory.
The same rules that apply to the "Common Folder" apply also to this folder.
**Please Note** - Your customers will **NOT** have the option to select a different "Auxiliary Folder" directory during the Setup process.

**Groups**
All files **must** be arranged in groups so you must create at least one group.
You add a Group by clicking the **[Add Group]** button.
More then one group is required if you plan to offer your customers the option to select "Typical", "Compact" or "Custom" installation.
**Groups are labeled with Green characters.**

**Folders**
Inside a group all files **must** be arranged in predefined folders.
You add a Folder by clicking the **[Add Folder]** button.

You may add the following predefined folders to each group:

- <Application Folder>    This is the directory you defined as the Application Folder.

- <Common Folder>    This is the directory you defined as the Common Folder.

- <Auxiliary Folder>    This is the directory you defined as the Auxiliary Folder.

- <Windows Directory>    This is the Windows directory as defined by the operating system.

- <System Directory>    This is the Windows\System32 directory as defined by the operating system.

- <System 16 Directory>    This is the Windows\System directory as defined by the operating system.

- <FontDir>    This is the Windows\Fonts directory as defined by the operating system.

- <MyDocumentsDir>    This is the "My Documents" directory as defined by the operating system.

- <RunTimeDir_1..8>    Up to 8 "Run Time" directories. A detailed description of those directories can be found at: "Directories".

When creating new Folders, you can also define the "Overwrite Files" policy for each folder individually.
The following options are available:

- **Setup Default** - Use the Global setting as defined on the "Switches" page.

- **Always** - The new file will always overwrite the old file.

- **When file is Newer or Same** - The new file will overwrite the old one only if it is newer or same.

- **When file is Newer** - The new file will overwrite the old one only if it is newer.

- **Never** - The new file will never overwrite the old one.

**Important** - The same Folders may appear in several different Groups.
**Folders are labeled with Red characters.**


**Directories**
You may create in each Folder any combination of Directories and Subdirectories.
The very same directory tree will be created on the target computer inside the target installation directory.
**Note -** Directories are not a must - you can add files directly inside Folders.
**Important** - The same Directories may appear in several different Folders.

By default all directories are marked with **BOLD Black** color.
If later on you set any attribute to this directory its color will changr to **BOLD Blue**.

## ADD

**Add Group**

Click the **[Add Group...]** button to add a new group to the setup.

- Enter the group name.
- Check the "Group MUST Always be Installed" Checkbox if you want this group to be installed always.
- Check the "Include Group in TYPICAL Installation" Checkbox if you want this group to be part of the TYPICAL installation.
- Check the "Include Group in COMPACT Installation" Checkbox if you want this group to be part of the COMPACT installation.

If you plan to offer your customers the option of CUSTOM installation, you may enter some descriptive text in the "Description" area, this text will help your customers decide if to install this group.

### Exclusivity Tag

Using the "Exclusivity Tag" you can force the user to select only one group out of several.

**EXAMPLE:**

You plan distribute your software to 3 countries: England, France & Germany.
You add to your software 3 help groups:
- Help English.
- Help French.
- Help German.

Now you want that if the user Check one of the 3 the others will be automatically UnChecked.

To achieve this goal do the following:
- Mark all 3 groups with the same **Exclusivity Tag** number (for instance 23).
- Check ONLY the "Help English" Group on the "CUSTOM Installation Page".

Now when your customer will Check the "Help French" group or the "Help English" group during Setup, the "Help English" group will be automatically UnChecked.

**IMPORTANT**

To make sure that one and only one group is ALWAYS selected you must set the **Exclusivity Tag** to a value greater then 100.

**Include Group in Setup by Serial Number level**

As previously explained, groups gives your customer the ability to control the content of the installation at setup time.
- By selecting COMPACT the user will normally install a smaller subset of the entire setup.
- By selecting CUSTOM the user will have a the option to manually select what to include in the actual setup.

The **Include Group in Setup by Serial Number level** offers you - the developer - the option to control what part of the setup will be installed on your customer's computer based on the Serial Number you provide him.

To take advantage of this option do the following:
- Decide how many Installation levels you want to use (upto 8 levels).
- Check in each group the levels you want this group to be included in.
- In the "Dialogs" page **Check** the "User Information" dialog and **UnCheck** the "Setup Type" and "Custom" dialogs.
- When you provide a serial number to your customer make sure you provide him/her with a serial number that starts with a number (1..8) or a character (A..H) according the level of installation you are offering him.

**EXAMPLE**
Lets say your setup include 4 groups: Group-1, Group-2, Group-3 and Group-4.

You plan to have 3 levels of installation:
- Level-1 that includes the groups:  Group-1, Group-2, Group-3 and Group-4.
- Level-2 that includes the groups:  Group-1 and Group-3.
- Level-3 that includes the groups:  Group-1 and Group-4.

To adjust this plan you will mark the Groups a follows:
- Group-1 included in Level-1, Level-2 and Level-3.
- Group-2 included in Level-1 only.
- Group-3 included in Level-1 and Level-2.
- Group-4 included in Level-1 and Level-3

When providing a Serial Number to your customer use the following samples:

- If JOHN is entitled to Level-1 Setup send him a Serial Number like: **1**593-7284 or **A**K8E-48DS.

- If MARY is entitled to Level-2 Setup send her  a Serial Number like: **2**593-7284 or **B**K8E-48DS.

- If DAVE is entitled to Level-3 Setup send him a Serial Number like: **3**593-7284 or **C**K8E-48DS.

The actual length and content of the serial number is not important. The only thing that matters is the first character that must be **1..8** or **A..H**.
This scheme works for all types of Serial Numbers that QSetup supports (DLL Based, Predefined & Tokenized).
To learn more about Serial Number click "Dialogs".

**Space Required on Drive (KB):**
By default QSetup will calculate the space required for every group on the target machine and present it on the Custom dialog.
You can manually set a different value in this edit box.

**Add Folder**
Click the **[Add Folder...]** button to add a new folder to the setup.

- Select one of the 4 predefined folders.

- Define the "Overwrite Files" level for the selected folder. If you select the first option (Setup Default) then the Overwrite level will follow the global definition on the Composer's **Switches** page.

- Define the "UnInstall Remove Policy" of the folder. Read the next paragraph (Add Directory) for explanations.

**Add Directory**
Click the **[Add Directory...]** button to add a new directory to the setup.

- Enter the new directory name. You may also select a name form the ComboBox, if you want to duplicate one directory in more then one group.
  - If required click the "FORCE Remove Directory During UnInstall" Checkbox.

  **Explanation** - During UnInstall the UnInstall program will attempt to delete all the files that the Setup program placed in the said directory. If the directory is empty the UnInstall program will delete it. If the directory is not empty because other files were added later to this directory, the directory will not be deleted.

- If the "Force Remove Directory During UnInstall" is checked, the UnInstall program will DELETE all the files in this directory and in all the subdirectories that are included in this directory, and then REMOVE this directory completely.

- If required click the "DON'T Remove Directory AND ITS CONTENT During UnInstall" Checkbox.

  **Explanation** - If a directory is marked with this option, the UnInstall program will not remove it or any subdirectory that starts from this directory. Also all the files in this directory and any subdirectory of this directory will not be erased.

- If required click the "DON'T Remove JUST Directory During UnInstall" Checkbox.

  **Explanation** - If a directory is marked with this option, the UnInstall program will not remove this directory, however ALL the contents of this directory (Files & SubDirectories) will be removed,

  **Important -** It is your responsibility to refrain from conflicts. A conflict may arise when one directory is marked as "FORCE Remove" and a subdirectory of it is marked as "DON'T Remove". In case of conflict, the "FORCE Remove" will prevail.

If you check any of the above mentioned switches the name of the directory will appear in **Blue** color.


### Add Files
Click the **[Add Files...]** button to add one or more files to the setup.
Select the files to add to each of the Folders you defined.
Files may be added directly into a Folder or inside Directories.

- By default all files are marked with **Black** color.


### Add File Tree
Click the **[Add File Tree...]** button to add tree of directories and the files included in them to the setup.
You may add a file tree to the setup files by selecting a certain directory in your computer. The result will be adding all the files and directory structure that starts from this directory.

- By default all files are marked with **Black** color.

**Link**
Using this option you can instruct QSetup to link the content of a directory to the setup delivery.
During compilation QSetup will scan the linked directory and add to the setup all the files and/or subdirectories that matches the link definition.

QSetup support 3 link options:

**Link Files By Mask..**.
RightClick the Files TreeView and select "Link/Link Files By Mask...".
The program will display a special dialog.
Click the **[Browse...]** button and select a file.
The program will create a path for a file in the form **\*.ext**.
During compilation QSetup will add all the files found in that directory that match the specified extention.

**Link All Files...**
RightClick the Files TreeView and select "Link/Link All Files...".
The program will display a special dialog.
Click the **[Browse...]** button and select a directory.
The program will create a path for a file in the form **\*.\***.
During compilation QSetup will add all the files found in that directory.

**Link Directory Tree...**
RightClick the Files TreeView and select "Link/Link Directory Tree...".
The program will display a special dialog.
Click the **[Browse...]** button and select a directory.
The program will create a path for that directory.
During compilation QSetup will add all the files and subdirectories found in that directory.
During Setup a simmilar directory tree will be built on the target machine.

# MISCELLANEOUS

**Check Dependency**
When you deliver your program to a customer you must make sure you also deliver all the DLLs and OCXs files that your program depends on.
The "Check Dependency" option will help you find out what DLLs and OCXs are required.

After you click the **[Check Dependency...]** button the program will check all the relevant files in your program (.EXE, .DLL & .OCX), and produce a list of all required files. Click **[Add Files]** to add the new files to your installation delivery. The files will be added to the current directory, you can then move them to another directory using Drag & Drop.

You can also check dependency for a single file by Right Clicking this file.

Files that were added using this option will be UnderLined in the files view.

"Check Dependency" is especially important if you program with "Visual Basic".

For more information Click: Check Dependency.

**Predefined Directories & <RunTimeDir_#>**
**QSetup** includes a long list of predefined directories like <Program Files Dir>, <WinDir> and upto 8 directories whose location is found at runtime. A detailed description of those directories can be found at: "Directories".
Normally, predefined directories are used as target starting directories for your installation.

**Drag & Drop**
You can use Drag & Drop for the following operations:
- Drag files from Windows Explorer to the Composer file tree area.
- Drag a directory tree from Windows Explorer to the Composer file tree area.
- Move files & Directories inside the Setup file tree.

**Target Executable**
Enter in this field the name of the main executable file of your program.
Use the "Browse" button to select one of the files in the Composer file tree.
This name will be used for the following tasks:
- Create shortcut(s) to your application.
- Register your program as an "Application" in the Windows Registry.
- Offer the user to launch the program at the end of the setup process.
**Note -** The name must include relative path and extension.

# Register As

**Shared file**
Registering a file as a shared one started with the introduction of Windows 95. The idea was to register shared DLLs, later on the concept evolved to include all type of files.

When registering a file as shared the Setup program will create a special entry for this file in the following Registry key:

HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\SharedDLLs
In this entry the Setup program will create a counter and set it to ONE.
If such an entry already exists the Setup program will increment the counter.

When UnInstalling the application the UnInstall program will decrement the counter, and proceed to delete the file only if the counter is set to ZERO.

You will normally use this option ONLY if you install a file that might be used by other applications.

**OCX/COM Server**
ActiveX files need to be registered under Windows.
Usually you perform the registration manually using the command line program RegSvr32.exe.
Use this option to instruct QSetup to perform this operation automatically during Installation.

> **NOTE** - This option is also available from the "Execute" page.

**Auto Test OCX/COM Server**
To be able to be registered as a Server a DLL or OCX must implement the "DllRegisterServer" function call. By default QSetup will check for every OCX or DLL if this function call is implemented and Enable the **OCX/COM Server** CheckBox accordingly.

To disable this automatic test do the following:
1. RightClick the "Files" TreeView pane.
2. Select the "Register As" menu.
3. Click the "Auto Test OCX/COM Server" menu.

**TLB**
TypeLibrary files need to be registered under Windows.
Use this option to instruct QSetup to perform this operation automatically during Installation.
TypeLibrary files are files with the following extensions: *.TLB.

**Font**
Font files need to be registered under Windows.
Use this option to instruct QSetup to perform this operation automatically during Installation.
Font files are files with the following extensions: *.TTF, *.FON, *.FOT, *.FNT.

**.NET Assembly**
Some .NET Assemlies (.dll) files need to be registered under .NET.
Usually you perform the registration manually using the command line program RegAsm.exe, which comes with the .NET Framework installation.
Use this option to instruct QSetup to perform this operation automatically during Installation.

# File Attributes

Normally when QSetup copies files during installation it clears all attributes of the copied files.
Use this option to set any combination of the following attributes:
- Read Only.
- Hidden.
- Archive.
- System.

**NOTE** - Every file that is marked as ReadOnly by the Setup program, WILL BE REMOVED during UnInstall. files that will be marked manually by the user WILL NOT BE REMOVED, and the UnInstall program will issue a corresponding message.

# Exclude From

QSetup can create setup delivery that will update automatically from the Internet. To learn more about this concept go to the "Auto Update" page.

**Regular Setup**
Any file or directory marked with this CheckBox will NOT be included in a Regular Setup.

**Auto Update**
Any file or directory marked with this CheckBox will NOT be included in an Auto Update Setup.

**UnInstall**
Any file marked with this CheckBox will NOT be UnInstalled during the UnInstall process. You must also make sure that any of the directories this file is included in is NOT marked as:
**"FORCE Remove Directory During UnInstall".**

**95, 98, ME, NT, 2K, XP, 2K3, Vista**, etc.
Any file or directory marked with any of these CheckBoxes will NOT be installed when installation is carried out on the relevant operating system.

**SFX File**
Any file or directory marked with this CheckBox will NOT be included in the Self Extract (SFX) file produced during compilation. For more information on **Non SFX** Files goto the "Project" page.
**[Clr All]** Click this button to clear all "Exclude From SFX File" marks in your project.

**Partial Setup**
"Exclude From Partial Setup" is only relevant if you create a Setup on CD, and mark the "Setup Type" option in the "Dialogs" page as "CD Setup". For more information on "CD Setup" goto the "Dialogs" page.
Any file or directory marked with this option will NOT be included in the "Partial Setup" when the user selects this option during installation.
"Exclude From Partial Setup" is only relevant for files that are marked as "Exclude From SFX File".

**IMPORTANT**
When you mark a directory with any of the above mentioned switches, this setting will influence also the content (Files and SubDirectories) of this directory

# Overwrite File

Using this option you can decide what the setup program will do when an old file is about to be replaced by a new one.
The following options are available:
- **Folder Default** - Use the setting as defined for the current folder.
- **Always** - The new file will always overwrite the old file.
- **File is Same or Newer** - The new file will overwrite the old one only if it is newer or same.
- **File is Newer** - The new file will overwrite the old one only if it is newer.
- **Never** - The new file will never overwrite the old one.

**IMPORTANT**
The "Overwrite File" policy may be defined in 3 places:
1. Per File.
2. Per Folder.
3. A global definition on the "Switches" page.

"Per File" has first priority.
If "Per File" is set to "Folder Default" then "Per Folder" is used.
If "Per Folder" is set to "Setup Default" then the "Global" definition is used.

**Note** - QSetup will attempt to compare the two files based on the file version. If version stamp cannot be found in the files, comparison will be made based on Date & Time of the files.

# ToolTip

Normally all files and directories in the TreeView display will be labeled in **BLACK** color.
Every file or directory that has any of its attribute set will be labeled with **BLUE** color.
Every file that was added to the list using the "Dependency Check" mechanism will be labeled with **RED** color.
When you move the mouse pointer over a **BLUE** or **RED** labeled file or directory, a tooltip will appear under the mouse pointer. This tooltip will include a list of all the attributes that were set for this file or directory.

# Modify Drive To

This option is useful when you need to move your source files to another drive.
Perform the following sequence of operations:
- Highlight all the files you want their drive to be modified.
- RightClick one of the selected files.
- Click the "Modify Drive To" item in the menu.
- Select the new drive.
   All the selected files will have their drive modified accordingly.

**IMPORTANT**
- All the selected files must reside on the same drive.
- The program will not check if the files are actually available on the new drive.

# Find & Replace

This option is useful when you need to modify the PATH of your source files.
To open the "Find & Replace" dialog, RightClick the Files TreeView and select "Find & Replace".

Enter the Path to Find in the first EditBox.
Enter the Replacement in the second EditBox.
Click the **[Replace]** button.

You can set the **Scope** of files to be modified by selecting:
- Global.
- Selected Lines.
- Current Line.

You can define what lines will be replaced:
- Always - Replace all lines that matches the "Path to Find" definition.
- Only when new file exists - The line will be replaced only if actual file exists in the new path.

**Undo**
Use the **[Undo]** button to restore the files to its status before the replacement.
Every click will take you one step back.
The undo option is valid only as long as you did not close the "Find & Replace" dialog.

# Find

This option is useful when you need to search for a speciifc file or path in the files list.
To open the "Find" dialog, RightClick the Files TreeView and select "Find".

# Path Alias

The Path Alias concept will help you when you need to provide a setup with similar structure but different set of files, or when you constantly need to move your development environment among several computers in your network.

To use the "Path Alias" option Check the **Path Alias** CheckBox at the bottom/right of the "Files" page.

When "Path Alias" is checked a special Panel will appear at the bottom of the "Files" page.
On this panel you will find 3 ComboBoxes marked as **<PATH_1>**, **<PATH_2>** & **<PATH_3>**.
In each ComboBox you can set a different directory path using the [Add...] button.

Now you can enter the alias **<PATH_1>** into any path of your source files and at Compile time this alias will be replaced with the content of the **<PATH_1>** ComboBox.

**EXAMPLE**
Let's say that your project include the following 2 files: "SimpleTest.exe" & "Service.dll".
You alternatively develop your project from home and from the office.

At home the files are located at:
- C:\Programs\Test\SimpleTest.exe
- C:\Programs\Test\Service.dll

In the office the files are located at:
- F:\Development\Tools\Test\SimpleTest.exe
- F:\Development\Tools\Test\Service.dll

To take advantage of the **Path Alias** option, you will set the path to the 2 files as follows:
- <PATH_1>\Test\SimpleTest.exe
- <PATH_1>\Test\Service.dll

When you are working from home you will set **<PATH_1>** to: "C:\Programs".
When you are working from the office you will set **<PATH_1>** to: "F:\Development\Tools".

**IMPORTANT**
The only way to add <PATH_#> to an existing path is by using the "Find & Replace" option.
To start "Find & Replace", highlight the relevant files, RightClick the files and select "Find & Replace".

## Other Files

When creating a setup you might need to apply "Path Alias" also to the following files:
- Image & Music files on the "Display" page.
- License Readme & Image file on the "Dialogs" page.
- Service files on the "Execute" page.
- Image & Text files on the "Billboard" page.

To apply the proper "Path Alias" you will either need to modify the text manually or highlight the relevant lines in a ListView, RightClick and select "Find & Replace".

## Commandline Parameters

You can also start the composer with a command-line simmilar to the following:
    /PATH_1=X:\MyPath
This will force PATH_1 to be X:\MyPath.

The same applies also for /PATH_2 and /PATH_3.

# Dialogs

The Setup program is actually a wizard that contains several dialogs, with each dialog handling another aspect of the Setup process. Use this page to select what Dialogs the user will encounter during the Setup process. The order of the Dialogs in the list is also the order of their appearance.
Click the **[Show Dialog]** button to see the selected dialog.

### Order of Dialogs

The order of the Dialogs in the list is also the order of their appearance in the Setup program. You can change the order of the dialogs by using Drag&Drop operation or by using the Arrow buttons.

Right Click the Dialogs list and select **Restore Default Order** to restore the original order of the dialogs.

**Note -** It is your responsibility to make sure that the new order does make sense, don't put the "Destination" dialog after the "Copy Files" dialog etc...

### Add Image
Click this button to add a unique Image to the selected dialog.
The recommended maximum size of the bitmap is:
- Classic:   Height=256, Width=120.
- Modern:   Height=307, Width=160 for the Large Image - Height=52, Width=52 for the Small Image.

The image must be of a Bitmap type (*.BMP).
You can add the same image to several dialogs.
You don't need to add the bitmap file to the files list, the Composer will add it automatically.

### Remove Image
Click this button to remove the image from the selected dialog.
The selected dialog will use the default image as defined in the "Display" page.

### License Agreement Text File
If you select the "License" dialog, you must provide text for this dialog. The text must be supplied in the form of a pure ASCII text file (Not UNICODE) or RichText file (.RTF).
Enter here the full path of the source text file.
**Note** - If you want this file to be kept on the target computer after installation, you must add it to the installation files list also.

### Readme Text File #1
If you select the "Readme" dialog, you must provide text for this dialog. The text must be supplied in the form of a pure ASCII text file (Not UNICODE) or RichText file (.RTF).
Enter here the full path of the source text file.
**Note** - If you want this file to be kept on the target computer after installation, you must add it to the installation files list also.

### Readme Text File #2

If you select "Propose to Display Readme file #2 after Installation" you must provide text for this dialog. The text must be supplied in the form of a pure ASCII text file (Not UNICODE) or RichText file (.RTF). Enter here the full path of the source text file.
**Note** - If you want this file to be kept on the target computer after installation, you must add it to the installation files list also.

### Use Notepad
Normally QSetup will display ReadMe file #2 in a window of its own. You can instruct QSetup to use Notepad for this purpose by checking the "Use Notepad" checkbox. If the file you are providing is an RTF file then WordPad or WinWord will probably be used.

### Space Required on Drive (MB)
QSetup will display on the "Destination" dialog the space required on the destination drive. QSetup calculates this data by reading the size of the files to be installed. If your installation also includes files that are copied using commands from the "Execution" Page, then QSetup will not be able to calculate their size. In such a case you will have to do the calculation manually. When entering the data you must set it in MegaBytes.

## Welcome Dialog

### Show recommendation to Close all Programs
Most Windows setup programs, displays a recommendation to close all programs before proceeding with the Installation process. We believe this is not always necessary, especially not for simple installations. If you feel this recommendation is necessary, Check the **Show recommendation to Close all Programs** Checkbox.

### Show Adware Disclaimer
If you are using Advertisement technology over the Internet, Check this option to bring this fact to the attention of your customers.

### Show Copyright Warning
Almost all setup programs display a standard **Copyright Warning** on the Welcome dialog. However if your installation does not include copyrighted material, you can hide this warning by UnChecking this option.

## User Information Dialog

In this dialog the user will be prompted to enter the following data:
- User Name.
- User Company Name.
- Software Serial Number.

The information will be stored by the Setup program in the registry under the following key:
HKEY_LOCAL_MACHINE\Software\<Company>\<AppName>\<Version>
Using the following Value Names:
- Name.
- Company.
- Serial.

### Request User Name
Check this item so that the user will be prompted to enter his name during installation.

**Request Company Name**
Check this item so that the user will be prompted to enter his company name during installation.

**Request Serial Number**
Check this item so that the user will be prompted to enter the Serial number of the software during installation.

- **NOTE** - At least one of the 3 items must be checked.

**Mandatory**
Each one of the above 3 items may be checked as "Mandatory". When "Mandatory" is checked the user will have to enter a value in the relevant field before proceeding.

**From PC Only**
"User Name" and "Company Name" may be checked as "From PC Only".
When this option is checked the relevant name will be read from the PC leaving the user no option to alter it during the setup process.


# Checking for Serial Number

You have 3 options available to check the validity of the Serial Number entered by your customer.
1. Provide a DLL that will perform the check.
2. Embed in your Setup program one or more predefined numbers.
3. Tokenized serial number.

**Serial Check DLL File**
Enter here the full path of your DLL.
A sample of the code required for the DLL is found at the end of this help page, or in the directory: **SampleDLLs** located under the QSetup directory.
Use the **[Test DLL...]** button to test your DLL.

**Predefined Serial Number(s)**
If you select the option of predefined numbers, you can use any combination of characters, numeral and punctuation marks - except for the **;** mark.

If you want to enter more then one predefined number you must use the **;** sign as delimiter:
the line: **abc123;tony2000;X#67\*** - will yield 3 serial numbers:
1. **abc123**
2. **tony2000**
3. **X#67\***

**IMPORTANT -** For added security the Predefined numbers are stored in your registry, not the QSP file.

## Tokenized Serial Number

The idea behind this option is to provide your customer with a serial number which is based on a secret Token provided by you, and a "User Name" and/or "Company Name" provided by your customer. This way you can provide every one of your customers with a unique "Serial Number".

Select this option and enter a token of several characters. Make sure you keep this token in secret. Later on when a customer buys your software, ask him to send you - by Email or any other means - his "User Name" and "Company Name".

When you receive this information, click the **[Create...]** button to open the "Create Serial Number" dialog.
Enter in this dialog the "User Name" and "Company Name" you just received and click **[Create Serial No.]**.
The program will create a unique serial number (Based on the "User Name", "Company Name" and the Token).
Send the new "Serial Number" to your customer - by Email or any other means - so that he can complete the installation process.

When creating a serial number you can select the following options:

**Group Level**
Select here "Level-1".. "Level-8" or "No-Level".
For more information on this option read about **Include Group in Setup by Serial Number level** in the "Files" page.

**Characters**
- Digits Only - The number will include ONLY digits.
- Text Only - The number will include ONLY Uppercase English characters.
- Text + Digits - The number will include Digits and Uppercase English characters.

**Length**
Set the length of the serial number (6 to 30 characters).

**Add Hyphens**
If this option is checked then hyphens will be added to the serial number every 3 or 4 characters. Hyphens are not counted in the **Length** of the Serial Number.

**Case Sensitive**
If this option is NOT checked then the program will ignore the case when entering "User Name" and "Company Name". We recommend that you leave this option UnChecked.

**Copy to Clipboard**
When this option is checked then as you click the **[Create Serial No.]** button, the program will place the newly created Serial Number in the clipboard. Goto your Email program and paste the new number directly to the Email message you are about to send to your customer.

**IMPORTANT**
- When you create a serial number you must use the SAME Token you used when you created the Setup delivery.
- For added security the Token is stored in your registry, not the QSP file.

## Setup Type Dialog

In this dialog the user will have the option to select one of 2/3 Setup Types.
- Typical.
- Compact.
- Custom (Optional).

**Show Compact Setup Also**

Check this Checkbox if you want the user to have the option to perform Compact installation.
**Note -** For this switch to be relevant you must mark only some of the Groups as included in compact installation.

**CD Setup**

This option is relevant if you create a setup that will be distributed on a CD.
If "CD Setup" is checked, the user will have the option to select one of 2 Setup Types:
- **Complete** - The entire program will be copied from the CD and installed on the Hard Disk.
- **Partial** - Only some files will be copied from the CD to the Hard Disk during installation. The user will have to insert the CD to the CD-Drive every time he runs the program. The files that are not copied are those marked as "Exclude From Partial Setup" on the "Files" page.

**Force Partial setup**

Check this option if you create a "CD Setup" and you want it to be installed as "Partial" - No Selection by the user.

## Copy Files Dialog

This dialog is for display only. It will show up during the copy files stage of the installation, and display the names of the files and a progress bar that reflects the progress of the copy process.
If you define a Billboard then this dialog will be replaced with a small progress bar at the bottom of the screen.
If you uncheck this dialog, the copy process will be performed without the display of this dialog, (this option may be reasonable if you install very few files).

**Show Progress Bar #2**

Usually QSetup displays only one progress bar during the "File Copy" process.
This Progress Bar merely counts the files as they are copied.
You may add another Progress Bar by checking the **Show Progress Bar #2** CheckBox.
The second Progress Bar is used for indication of the Copy Process of every individual file.
We suggest that you add this Progress Bar only if you deliver VERY LARGE files in your setup.

**Perform Silent Setup**

When this option is checked, the Setup program will run without intervention of the user.
The Setup program will skip all the first pages, go straight to the "Copy Files" dialog, perform the installation and close.
You can achieve the same effect by unchecking all the dialogs except the "Copy Files" dialog.
You can achieve the same effect by adding **/silent** parameter on the command-line of the setup program.

**Enable Cancel Button**
Check this option if you want to give the end user the option to abort a "Silent Setup".

**Perform Hidden Setup**
When this option is checked, the Setup program will run completely in the background, presenting no dialog during the process.
You can achieve the same effect by adding **/hide** parameter on the command-line of the setup program.

**IMPORTANT**
**/silent** & **/hide** will have no effect if the "User Information" dialog is selected.

# Complete Dialog

**Propose to Restart the Computer after Installation**
Check this option if you want the user to restart his PC after Installation. Setup will ask the user for confirmation.
**FORCE** - If this option is checked, setup will always restart the PC - no questions asked.
**IMPORTANT** - If setup fails to copy one or more files, setup will mark them for later copy and prompt the user to restart his PC - even if this checkbox is not checked.

**Propose to Launch the Application after Installation**
Check this option if you want the user to launch the Target Executable right after Installation. Setup will ask the user for confirmation.
**CHECKED** - Check this option to set the default status of the relevant checkbox to **checked**.

**Propose to Display Readme file #2 after Installation**
Check this option to display a readme file after installation. Setup will ask the user for confirmation.
**CHECKED** - Check this option to set the default status of the relevant checkbox to **checked**.

# Serial Check DLL File

If you want your customers to enter a serial number during installation, you must provide a special DLL or "ActiveX DLL" to check the validity of the entered serial number.
Sample code for the DLL can be found in the directory **SampleDLLs** under the QSetup directory.

The DLL or "ActiveX DLL" must export two functions with the following prototype:

**C/C++ Code:**
```
__declspec(dllexport) int __cdecl GetDllVersion()
__declspec(dllexport) int __cdecl GetSerialOK(HWND Wnd, char*
Serial, char* User, char* Company)
```

**Pascal Code:**
```
function GetDllVersion: integer; cdecl;
function GetSerialOK(Wnd: HWND; Serial,User,Company: PChar):
integer; cdecl;
```

**Visual Basic Code:**
```
Public Function GetDllVersion() As Integer
Public Function GetSerialOK(Wnd As Long, Serial As String, User As
String, Company As String) As Integer
```

**C# Code:**
```
public int GetDllVersion()
public int GetSerialOK(int Wnd, string Serial, string User, string
Company)
```

**VB.NET Code:**
```
Public Function GetDllVersion() As Integer
GetSerialOK(ByVal Wnd As Integer, ByVal Serial As String, ByVal
User As String, ByVal
Company As String) As Integer
```

The return value of the first function must be **1**.
The return value of the second function must be **0** if the test failed or **1** if the test
succeeded.

**Sample C/C++ Code for the DLL** (Compiled with Visual C++ 6)

```
#include "stdafx.h"
#include <string.h>

extern "C" {
__declspec(dllexport) int __cdecl GetDllVersion()
{
    return 1;
}
}

extern "C" {
__declspec(dllexport) int __cdecl GetSerialOK(HWND Wnd, char* Serial,
char* User, char* Company)
{
    if (strcmp("1234",Serial)==0)
        return 1;
    else
        return 0;
}
}
```

**Sample C/C++  Code for the DLL** (Compiled with Borland C++ Builder)

```
#include <string.h>

__declspec(dllexport)  int __cdecl GetDllVersion()
{
    return 1;
}

__declspec(dllexport)  int __cdecl GetSerialOK(HWND Wnd, char* Serial,
char* User, char* Company)
{
    if (strcmp("1234",Serial)==0)
        return 1;
    else
        return 0;
}
```

### Sample PASCAL Code for the DLL (Compiled with DELPHI 6 & DELPHI 7)

```pascal
library SerialCheck;

uses
  Windows,SysUtils;

function GetDllVersion: integer; cdecl;
begin
  Result:=1
end;

function GetSerialOK(Wnd: HWND; Serial,User,Company: PChar): integer;
cdecl;
begin
  if (StrComp('1234',Serial)=0) then
    Result:=1
  else
    Result:=0;
end;

exports
  GetDllVersion,
  GetSerialOK;

begin
end.
```

### Sample "Visual Basic" Code for the "ActiveX DLL" (Compiled with VB6)

```vb
VERSION 1.0 CLASS
BEGIN
  MultiUse = -1  'True
  Persistable = 0  'NotPersistable
  DataBindingBehavior = 0  'vbNone
  DataSourceBehavior  = 0  'vbNone
  MTSTransactionMode  = 0  'NotAnMTSObject
END
Attribute VB_Name = "SerialCheck"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = True
Attribute VB_PredeclaredId = False
Attribute VB_Exposed = True

Public Function GetDllVersion() As Integer
  GetDllVersion = 1
End Function

Public Function GetSerialOK(Wnd As Long, Serial As String, User As
String, Company As String) As Integer
  If (Serial = "1234") Then
    GetSerialOK = 1
  Else
    GetSerialOK = 0
  End If
End Function
```

### IMPORTANT Note for VB Programmers
The VB_Name attribute MUST BE: "SerialCheck".
For instructions on how to compile the VB code please read the file
"HowToCompileSamples.txt" found in the VB directory.

**Sample C# Code for the DLL (Compiled with VS)**

```csharp
using System;
using System.Runtime.InteropServices;

namespace QSetup
{
    [ClassInterface(ClassInterfaceType.AutoDual)]
    public class SerialCheck
    {

        //----------- GetDllVersion -----------

        public int GetDllVersion()
        {
            return 1;
        }


        public int GetSerialOK(int Wnd, string Serial, string User,
string Company)
        {
            if (Serial == "1234")
                return 1; // OK - Serial Number is correct.
            else
                return 0; // Failure - Serial Number is wrong.
        }
    }
}
```

**Sample VB.NET Code for the DLL (Compiled with VS)**

```vbnet
Imports System
Imports System.Diagnostics
Imports System.Runtime.InteropServices

Namespace QSetup

    Public Class SerialCheck

        Public Function GetDllVersion() As Integer
            GetDllVersion = 1
        End Function


        Public Function GetSerialOK(ByVal Wnd As Integer, ByVal Serial
As String, ByVal User As String, ByVal Company As String) As Integer
            If (Serial = "1234") Then
                GetSerialOK = 1
            Else
                GetSerialOK = 0
            End If
        End Function

    End Class
End Namespace
```

# Switches

## Operating Systems

Select the operating systems your program was designed to work with.

Starting from version 10.0.0.0 QSetup supports the following operating systems:
**32-bit:** 95,98, ME, NT, 2K, 2K3, XP, Vista, Win7, Win8, Win10.
**64-bit:** XP.64, Vista.64, Win7.64, Win8.64, Win10.64.

**Create 64-bit Setup**
The switch "Create 64-bit Setup" controls the 32/64-bit setup state.
When this switch is unchecked (default) a setup produced with QSetup can be installed as a 32-bit application on all the supported 32-bit operating systems.
When this switch is checked the setup produced by QSetup can be installed only on a 64-bit operating system.

The setup state can be changed at setup time using the following "Execute" commands:
- Set 64 Bit State.
- Set 32 Bit State.
- Restore Original Setup State.

Using these "Execute" commands a 32-bit setup can access 64-bit resources and vice-versa.

## Register as Application

Following Microsoft recommendations, every application should register itself under Windows by adding the following 2 keys to the registry:

HKEY_LOCAL_MACHINE\Software\CompanyName\ProgramName
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\App Paths\ProgExe

If you supply a very simple application or you just supply a collection of data files (images, documents etc...) you can refrain from adding those keys to the registry by UnChecking "Register as Application" CheckBox.

## Register project in the HKLM\Software branch

By default every project will add the following key to the registry:

HKEY_LOCAL_MACHINE\Software\CompanyName\ProgramName

In this key the following values will be written:
Version
ProgName
InstallDir
UnInstallString

You can use this option to diable this operation, however we recommend that you will leave this option enabled at all times.

## Create Setup.log File

If this option is checked then the Setup program will create a log file with the name "Setup.log" during the installation process. The file will be stored in the Application Folder.

## Extract Files

Request Confirmation before Extract
When this option is checked then your customer will be presented with a special message box right after starting the setup file. This message box will display information about the free space available on the TEMP drive and the space required for extracting the setup file. Your customer will also have the chance to abort the installation at this early stage.

**Show Activity Bar**
This option is important if you provide a very large setup that will take long time to extract. When this option is checked a small additional progress bar will be displayed on the Extraction Dialog. This bar will provide a better activity indication.

## Previous Installation

**If previous installation found then:**

Check this option if you want to perform certain operations on a previous installation before the actual Setup process starts.

Select one of the following options:

- **UnInstall before setup starts**
  QSetup will uninstall a previous installation of the current project.
  This option is valid only if your setup was compiled with "Create Automatic UnInstall" checked.

  **Mode**
  You can select from 3 options on how to perform this specific UnInstall:
  - Normal - The UnInstall will execute in normal fashion displaying the UnInstall dialog and prompting the user to click the relevant buttons.
  - Silent - Display the UnInstall dialog but don't wait for any user input.
  - Hidden - Perform the UnInstall in the background no dialog and no user input.

  The MODE setting here will override any setting you do for normal UnInstall.

- **Confirm Continue Setup**
  QSetup will display a confirmation message asking the user to confirm installation of the new version. The user will have the option to Confirm or to Deny.

- **Inform and Continue Setup**
  QSetup will display an attention message, reporting the user that he is about to upgrade an old version. The user will not have the option to stop the installation.

- **Inform and Abort Setup**
  QSetup will display an attention message, reporting the user that an older version is installed and the installation process will abort.

- **Just Abort Setup**
  QSetup will abort the installation no message issued.

# Running Executable

It is always recommended to stop the running program before attempting Setup or UnUinstall.

### Test for Running Executable before Installation
Your customer may attempt to Install the program for the second time over a previously installed copy.
THIS IS A PERFECTLY LEGAL ATTEMPT.
However, it is recommended that the old executable is NOT running during the process.
If you check this option then the setup program will check at the start of the installation if the old executable is running and issue a warning message prompting your customer to close the program.
This option is valid only if the "Register as Application" option is checked.

### Test for Running Executable before UnInstall
If you check this option then the UnInstall program will check at the start of the UnInstall process if the old executable is running and issue a warning message prompting your customer to close the program.

# Run/RunOnce

RunOnce -   If you check this option the program will run automatically the next time your customer will reboot his PC.

Run -   If you check this option the program will run automatically every time your customer will reboot his PC.

# Administration

NT-Class machines (NT, 2000, XP, 2003, Vista, Win7, Win8, Win10) uses the concept of Regular-User vs. Administrator. Usually under an NT Machine a Regular-User has limited rights compared to an Administrator. In essence a Regular-User has limited access to the registry and to the file system. Consequently it is recommended that only Administrators will Install/UnInstall software on an NT-Class machine.

### Test for Administrator Right before Installation
When this CheckBox is Checked, QSetup will perform a check for Administrator rights at the very beginning of the Setup process.

If you supply a very simple application or some data files, you may UnCheck this CheckBox and let a Regular-User install your files to an NT-Class machine.
QSetup is designed to proceed with the installation even if some registry entries where not written properly, however QSetup will abort the installation if a failure occurred during creation of a new directory.

### Test for Administrator Right before UnInstall
When this CheckBox is Checked, QSetup will perform a check for Administrator rights at the very beginning of the UnInstall process.

**Ignore Administrator test on Vista and Up**
Starting from Vista, the Windows Operating System grants Administrator rights to every user during setup, so there is no need to perform the above two tests when your setup is running on Vista and up.
When this option is checked, Administrator tests on Vista and up will always return true.

**Allow Full Access to All Users**
The NT-Class operating system has the ability to limit user access to files and directories. Usually an administrator can control the access using the command-line program CACLS.EXE.
You can use this option to make sure that all users have FULL Access to all the files and directories that the Setup program has installed.
Of course to perform this operation the person who runs the setup program must have administrator right.

**IMPORTANT -** "Allow Full Access" is NOT available in the LITE Version.

# Overwrite Files

This switch will determine how the Setup program will behave when it attempts to overwrite a file during installation.
The following options are available:
1. **Always** - The new file will always overwrite the old file.
2. **When file is Newer or Same** - The new file will overwrite the old one only if it is newer or same.
3. **When file is Newer** - The new file will overwrite the old one only if it is newer.
4. **Never** - The new file will never overwrite the old one.

**Note** - This global definition may be overridden individually by any Folder and/or by any file.
**Note -** QSetup will attempt to compare the two files based on the file version. If version stamp cannot be found in the files, comparison will be made based on Date & Time of the files.

# Autorun

The Autorun option is needed if you plan to provide your setup on a CD.

**Create Autorun.inf File**
When this option is checked a file by the name "autorun.inf" will be created when you compile the setup. The file will be placed in the Project directory along with your Setup file. If you copy this file and the Setup file to the ROOT of the CD then the setup program will start automatically when the user insert the CD into the drive.

**Perform Autorun Test**
When this option is checked the Autorun option will work only as long as the program was not installed yet. Once the user installed the program, subsequent insertions of the CD will not start the setup procedure. When this option is checked, a small program by the name **AutoRunTest.exe** will be added to the project directory. You must copy this file also to the ROOT of the CD together with the **Autorun.inf** file and the setup file.

**Start Application if Already Installed**
When this option is checked the Autorun option will prompt the user to start the application when he/she inserts the CD into the drive and the program is already installed.

## .NET Framework

### Test for .NET Framework before Installation

Use this switch to test for the existence of **.NET Framework** on the target machine. If **.NET Framework** is not found the setup program will issue a corresponding message and abort.

You can perform a similar test also from the "Execute" page.

### Install .NET Framework

Use this option if you want to install .NET framework during the setup.
Specify here the full path of the DOTNETFX.EXE file.

If you are installing .NET Framework 3.0 then you can enter here the full path of: DOTNETFX3SETUP.EXE.
This setup file from Microsoft is only 2.8 MB in size and it will download and install the entire framework directly from Microsoft website.

### Install .NET Framework from the Internet

Use this option if you want to install .NET framework from the Internet during the setup.
Specify here the URL to download the DOTNETFX.EXE file from.
QSetup will download the file using HTTP Protocol.
Example: **http://www.microsoft.com/download/dotnetfx.exe.**

### [Verify URL]

Click this button to verify that the URL you specified for the DOTNETFX.EXE file is correct.

### Required Version Is In

Select here what .NET Framework versions you want to test for before installation.

For instructions on how to install .NET Framework using the "Execute Engine" read the following link:
www.pantaray.com/howto.html#howto-001

# Shortcuts

Use this page to define Shortcuts and links to your program, to be installed on the target computer.

IMPORTANT - For shortcuts to work properely we recommend that you define a "Target Executable" on the "Files" page.

## Start/Program Menu Shortcut

This is the main link to your program. This link is accessible to the user by clicking the "Start" button found on the Windows Taskbar and then selecting the "Program" menu.

**Main Shortcut Folder**
The name you enter here will be the name as appears on the Program menu.
Normally you will enter here the "Program Descriptive Name" as you entered in the "Project" page.
If you want your menu entry to include submenu add a Backslash and then the name of the submenu.
   **Example -** "Microsoft\Word for Windows".

**Add Main Shortcut**
The main shortcut is normally the one that start the Target Executable. If you do not supply an executable as your main file you may UnCheck this option.

**Shortcut Name**
The name you enter here will be the menu item that will actually launch the program when the user clicks it.
   **Example -** "Word 2000".

**Parameters**
Enter here the command-line parameters if required.

**Work Dir**
Enter here the target working directory. Use the "Browse" button to select the required directory.

**Run**
Select here the way the program will start when clicking the shortcut.
The options are: "Normal Window", "Maximized", "Minimized".

**Icon File**
Most EXE programs include internal icon. Use this entry if you install files that do not include an icon or you want  your program to be identified by another icon. The file you specify here must first be added to the general files list on the "Files" page. You can specify here any of the following File-Types: **.EXE**, **.DLL**, **.ICO**.

**Icon Number**
Icon files may include more then one icon. The icons are numbered from zero and up. Specify here the number of the icon you select.

# Other Start/Program Shortcut Items

You may add more Shortcut items like HELP or other services.
Shortcut items may be added to the Program menu or other locations.

Click **[Add...]** and then
Enter a descriptive name to the service in the **Shortcut Name** field.
Select the Shortcut location:
- Program Menu
- Desktop
- Start Menu
- Startup
- Send To
- Quick Launch
- My Documents

If "Program Menu" is selected then enter the folder name in the **Shortcut Folder** field.
Enter the name of the File or Folder associated with this service in the **File/URL/Folder** field.
Use the browse button to select a File or a Folder from the "Files" list.
Of course you must include the associated file with the installation files.

### Internet Shortcut
You can create a Shortcut to an Internet URL by specifying a valid URL in the **File/URL/Folder** field.
A valid URL must start with any of the following string characters:
- http:
- https:
- ftp:
- mailto:
- www.

### Hotkey
You may add Hotkey to your shortcut.
Check the **Hotkey** checkbox and select the key combination.

### Remove Shortcut During UnInstall
If you leave this option UnChecked then the Shortcut will not be removed during UnInstall.

### Include in Operating Systems
Select **All** to add the shortcut to all the operating systems, or select only the required ones.

### Exclude From
This option is usefull if you are creating a CD-Setup.
Using this option you can exclude a shortcut from "Complete Setup", "Partail Setup" or both.

## Other Shortcuts

Windows offer some other type of shortcuts.
You may offer your customers the option to add any or all of these shortcuts.
If the "Checked" CheckBox is checked - the relevant shortcut, as seen by your customer,
will be checked by default.

### Start Menu

The shortcut name you enter here will appear in the upper part of the "Start" menu.
Normally you will enter here the "Program Descriptive Name" as you entered in the
"Project" page.

### Desktop

The shortcut name you enter here will appear on the "Desktop".
Normally you will enter here the "Program Descriptive Name" as you entered in the
"Project" page.

### Send To

The shortcut name you enter here will appear on the "SendTo" popup menu in various
Windows applications.
Normally you will enter here the "Program Descriptive Name" as you entered in the
"Project" page.

### StartUp

The shortcut name you enter here will appear on the "StartUp" menu, and will run at PC
Startup.
Normally you will enter here the "Program Descriptive Name" as you entered in the
"Project" page.

### Quick Launch

The shortcut name you enter here will appear on the "Quick Launch" bar.
Normally you will enter here the "Program Descriptive Name" as you entered in the
"Project" page.

## Language Support

Click the button **[Language Support...]** to open a special "Shortcut Language Support"
dialog.
Using this dialog you can translate all the "Shortcut Names" & "Shortcut Folder Names" to
every language you want to support in your setup.
Using this dialog you can also select Files or URLs per language, thus you can for
instance create webpages in several languages and call the appropriate page according
to the setup language.

## Shortcuts are Available for

This option is useful for NT Class computers, that have the option to login different users
with different privileges. With this option you can select if the program will be available
only for the user who installed it or for all users of the said computer.
● If the user who installs the program does not have Administrator rights, then all the
shortcuts will be available for the "Current User Only".

### IMPORTANT

To properely test your multilingual setup you must adjust your operating system
to the language under test.
For more info read the following link: www.pantaray.com/language.html#testing.

### Selected by End User

If you check this option then your customer will have the option to select shortcuts availabiliy during setup time.

# UnInstall

**Add UnInstall Shortcu**t
If you check this Checkbox, the setup program will add an "UnInstallation Shortcut" to the Start/Program menu. For this item to be active, you must check the "Create Automatic UnInstall" Checkbox in the "Switches" page.

**Force Remove Main Shortcut Folder**
Normally the UnInstall program will attempt to remove the main shortcut folder.
If the user has added manually some more shortcuts to this Folder, or another installation program has placed shortcuts in the same Folder, the Folder will NOT be removed during UnInstall.
You can FORCE the UnInstall program to remove this folder by checking this option.

# Associate

In this page you will define Associations between Files of certain type and your Application. Every time you double-click an associated file from the Windows file manager - Windows will launch your application and instruct it to open the file you selected, by transferring its name on the command line.

To add Association item click the **[Add Item]** button.

The "Add Association Item" dialog will appear. In this dialog you will enter the following items:

**Association Name:**
Every Association item must have a unique name. This name is only used internally by Windows.
    **Sample:** "WordDoc"

**File Description:**
This will be the description of the file as appears in the file manager under the "Type" column.
    **Sample:** "Microsoft Word Document"

**File Extension:**
This is the extension of the files - including the leading period.
    **Sample:** "**.**doc"

**Program Descriptive Name:**
A descriptive name of the program that is associated with this file, and will be launched when you double-click a file of this type.
    **Sample:** "Word for Windows"

**Program Path:**
The full path of the associated program. Naturally this must be a program that you provide with the installation delivery.
Click the **[Browse...]** button to select the program.
    **Sample:** "<Application Folder>\Microsoft Office\Office\WINWORD.EXE"

**Icon Path:**
By providing an Icon definition you will instruct Windows to display this Icon along with the said file type.
You may enter here a file name with any of the following extensions: **.EXE, .DLL, .ICO**.
Naturally this file must be provided with the installation delivery.
Click the **[Browse..]** button to select it.
    **Sample:** "<Application Folder>\Microsoft Office\Office\WINWORD.EXE"

**Icon Number:**
EXE files and DLL files may include more then one Icon. The icons are numbered from zero and up. Specify here the number of the Icon you select.

**Enable**
After you have defined an Association Item You can disable its operation by UnChecking the Enable Checkbox. A disabled item will not be executed during Setup or UnInstall process.

## Open With

**What is Open With?**

Using OpenWith you can give your customers the option to open a file in your
application in a controlled manner without regular association.

Say you developed an editor called "SuperEdit" that can handle .HTML and .DOC files.
Now you can associate HTML and DOC files with your editor, but then every time the
user will double click an HTML or DOC file they will open in your editor - rather then the
Browser or WinWord, which are the natural applications for HTML and DOC files.

If you associate HTML and DOC files with your editor using OpenWith, then every time
the user will RightClick an HTML file or a DOC file he will see in the Context menu an
entry similar to this one: "Open With SuperEdit". Selecting this entry will open the file in
your editor.

To associate your application using OpenWith enter the following data items:

**Name:**
Enter here a simple name like "SuperEdit". This name is only used internally by the
registry. Select a name that is believed to be unique in the registry.

**Menu Text:**
Enter here the text that will appear in the context menu. The text can be something like
"Open With SuperEdit" or "Edit With SuperEdit".

**Extensions:**
Enter here the list of extensions that should be associated with your application using the
OpenWith option. Use comma (,) to separate the extensions. For our example the list will
look like this:
HTML,HTM,DOC

# Registry

In this page you will define Keys and Values that will be added to the system Registry during Setup and/or removed from the system Registry during UnInstall.

## Add Registry Item

To add a Registry item click the **[Add Item]** button.

When adding a Registry item you will need to enter the following data:

**Root**
Select any of the following:
HKEY_CLASSES_ROOT
HKEY_CURRENT_USER
HKEY_LOCAL_MACHINE
HKEY_USERS
HKEY_CURRENT_CONFIG

**Key**
Enter a registry key in a form similar to the following:
Software\MyCompany\MyApplication

**Value Name**
Enter the name of the data item

**Value Type**
When entering value you can use one of the following 5 value types:

- String - (REG_SZ)
  Enter any text.

- Integer - (REG_DWORD)
  Enter a string that represents a positive integer and include only the following characters: **0..9**

- Hex - (REG_BINARY)
  Enter a value in the following format: FF,00,08,12,1A - This string represents 5 bytes with the following values: 255,0,8,18,26. There is no limit on the number of bytes that may be entered in this way.

- MultiString - (REG_MULTI_SZ)
  Enter a sequence of several strings delimited with the alias <0>. When writing the data to the registry, QSetup will modify the alias <0> to char(0).
  <0> is built of 3 characters: LessThen + Zero + GreaterThen.

- ExpandString - (REG_EXPAND_SZ)
  Enter a string that may include environment tokens like %PATH% or %TEMP% or %SystemRoot%.
  Later on when those strings are read from the registry the system will automatically replace the tokens with actual data from the environment.

**Value**

Enter here the actual value data. The way you enter the data depends on the **Value Type**.

**During Setup**
Normally you will set this option to **Create**. which means create this Registry Key/Value during Setup.
You may also specify here **Create if not Exist**. which means create this Registry key during Setup - only if such a Key/Value does not already exist on the PC.

**During UnInstall**
Normally you will set this option to one of the following:
**Remove Key -** which means remove this Registry key during UnInstall.
**Remove Value -** which means remove this Registry value during UnInstall.

**Enable**
After you have defined a Registry Item You can disable its operation by UnChecking the Enable Checkbox. A disabled item will not be executed during Setup or UnInstall process.

## Import Reg File

A very effective way to add registry Keys & Values to your setup is by importing them from a *.REG file.

To create *.REG files do the following:
- Open REGEDIT.
- Highlight one or several keys you want to export.
- From the "Registry" Menu select "Export Registry File..."
- Enter a file name and Click "Save".

To Import the *.REG file click the **[Import Reg File]** button and select the file.
All the exported keys will be added to your setup.

## Install Reg File

Another very effective way to add registry Keys & Values to your setup is by installing a complete *.REG file.
Click the **[Add Item]** button and select the registry file you want to install.
IMPORTANT - a reg file can not be uninstalled.

# IniFile

In this page you will define Values that will be added to an INI file during Setup and/or Removed from the INI file during UnInstall.

## Add IniFile Item

To add an INI file Value item click the **[Add Item]** button.

**File Name**
Click the [Browse...] button to define the file name of the INI file.

**Section Name**
Enter the Section name.

**Value Name**
Enter the Value name.

**Data**
Enter the valus Data.

**During Setup**
Normally you will set this option to **Create**. which means create this IniFile item during Setup.

**During UnInstall**
Normally you will set this option to **Remove** or **Ignore**.

**Enable**
After you have defined an INI file Item You can disable its operation by UnChecking the Enable Checkbox. A disabled item will not be executed during Setup or UnInstall process.

# XML

In this page you will define Values that will be added/removed to/from an XML file during Setup and/or UnInstall.

## Add XML Item

To add an XML item click the **[Add Item]** button.

**File Name:**
Click the [Browse...] button to define the file name of the XML file.

**Xml Node Path:**
Define the full path of the node you want handle starting from the root node.
The elements of the path must be separated with a dot (.).

**Value:**
Enter the value you want to set for the node..

**Sample:**
   Node Path:    order.address.IP
   Value:        127.0.0.1

will yield the following XML structure in your file:

```
<order>
   <address>
      <IP>127.0.01</IP>
   </address>
</order>
```

**During Setup**
Normally you will set this option to **Create Value**. which means create this XML item during Setup.

**During UnInstall**
Normally you will set this option to **Erase Value** or **Ignore**.

**Enable**
After you have defined an XML Item you can disable its operation by UnChecking the Enable Checkbox. A disabled item will not be executed during Setup or UnInstall process.

## Defining an XML path

The sample **order.address.IP** is a very basic Node Path sample.
When using this sample with the command: **Create Value** a new path will be created if not exists or an existing value will be updated.
A new XML file will be created if not exists.

**The ROOT Node**

The Root Node (**order** in the above sample) can be replaced with the token **root**, thus the Node Path: **order.address.IP** is equvalent to **root.address.IP**.

### Node Duplication
The XML standard supports multiple nodes with the same name. If your XML file already contains one or more node path with the structure of **order.address.IP** and you want to add another node use the following Node Path:
**order.address,*.IP**
This Node path will duplicate the node **address** and add the node **IP** inside the **address** node.

### XML Attributes
The XML standard supports Attributes that can be added to a node.

### Sample:
Node Path:      order.address.IP^Location
Value:          "New York"

will yield the following XML structure in you file:

```
<order>
   <address>
      <IP Location="New York"/>
   </address>
</order>
```

**IMPORTANT**
The "Add XML Item" dialog contains a [Test...] button. Click this button to open a special testing dialog where you can test how your new definitions are reflected in your XML file.

# Example

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<CDCollection>
   <owner age="32" gender="male">
      <lastname>Murphy</lastname>
      <firstname>Steve</firstname>
      <address>Broadway 1</address>
      <city>Los Angeles</city>
      <phone>318-349-2485</phone>
   </owner>
   <CD tracks="13" released="1998">
      <singer gender="female">Madonna</singer>
      <title>Ray of Light</title>
   </CD>
   <CD tracks="12" released="1993">
      <singer gender="female">Gloria Estefan</singer>
      <title>Mi Tierra</title>
   </CD>
   <CD tracks="10" released="1990">
      <singer gender="female">Gloria Estefan</singer>
      <title>Abriendo Puertas</title>
   </CD>
   <CD tracks="14" released="1996">
      <singer gender="male">Joe Cocker</singer>
      <title>Organic</title>
   </CD>
```

```
</CDCollection>
```

**Reading the sample**
The following table show a collection of XML path definitions and their associated values.

| XML Path | Value |
| --- | --- |
| CDCollection.owner.lastname | Murphy |
| root.owner.firstname | Steve |
| CDCollection.owner^age | 32 |
| root.owner^gender | male |
| CDCollection.CD,1.title | Mi Tierra |
| root.CD,0.SINGER | Madona |

**IMPORTANT**
XML path definitions are not case sensitive. Thus the following 2 definitions will both yield the value: "Madona".
   root.CD,0.SINGER
   root.CD,0.singer

# Environment

In this page you will define variables that will be added to the Environment during Setup and/or Removed during UnInstall.

## Add Environment Variable

To add an Environment variable click the **[Add Item]** button.

### Create

Use this option to create a new variable. If a variable with this name already exists, it will be replaced with the new one.

### Append

Use this option to add data to an existing variable. If the variable does not exist it will be created. If the appended data is already found in the variable the operation will abort.
If you are appending to the PATH variable then QSetup will automatically handle semi-colons for you.

### Remove

Use this option to remove an existing variable. If the data was previously appended the variable will revert to its original status.

### Scope

Environment variables may be added to the **System** or to the current **User.**
Variables that are added to the **System** are visible to all users.
This option is valid only on NT-Class machine.

### Enable

After you have defined an Environment variable Item You can disable its operation by UnChecking the Enable Checkbox. A disabled item will not be executed during Setup or UnInstall process.

## Variable Expansion

Environment variables may be expanded by using the % sign.

For instance:
If your current PATH variable is:
```
C:\WINDOWS;C:\WINDOWS\SYSTEM
```

And you write a new PATH variable with the following value:
```
%PATH%;C:\MyProg
```

Then the final path will be:
```
C:\WINDOWS;C:\WINDOWS\SYSTEM;C:\MyProg
```

We strongly recommend that you use **Append** instead of **Expansion** as Appended variables can be safely removed during uninstall.

## NT vs 9X

When installing on an NT-Class machine (NT, 2K, XP, 2003, Vista, Win7, Win8, Win10) environment variables are handled using the registry and any change you make to the environment effects the operating system immediately.
When installing on a 9X-Class machine (95, 98, ME) environment variables are handled using the AUTOEXEC.BAT file, thus a reboot is required to effects the operating system.

# Properties

Most Windows applications include various types of resources. Normally resources are added to the application by the compiler when you compile your application.

Using this page you will have the option to define "Version Info" and "Icon" resources for the Media file that the "QSetup Composer" produce for you to send to your customers. The information you define here can later be viewed by watching the windows generic "Properties" dialog.

**IMPORTANT**
This page is valid ONLY if you are running the "QSetup Composer" on an NT-Class machine (NT, 2000, XP, 2003, Vista, Win7, Win8, Win10). However - any setup produced with this option will faithfully run on any Win32 machine.

## Version Info

When Version Info is properly set, your customer will see the correct information when he requests the "Properties Page" of the Media file (Right Click on the file from Explorer).

**Don't Modify Version Info**
If you select this option then the original version info, which was set in the file by "Pantaray Research Ltd." will be maintained.

**Copy Version Info from Target Executable**
If you select this option, the version info of the Target Executable will be copied to the Media file, every time the Media file is compiled.
This option is valid only if you selected a Target file and the file is EXE or DLL.

**Define Version Info Manually**
Select this option to set the "Version Info" manually.

**[Read Info From Target Executable]**
Click this button before you begin entering information manually. This will copy the data from the target executable to the list.
This option is valid only if you selected a target file and the file is EXE or DLL.

## Icon

Use this option to modify the Icon of the Media file.

**Don't Modify Icon**
If you select this option then the original icon, which was set in the file by "Pantaray Research Ltd." will be maintained.

**Copy Icon from Target Executable**
If you select this option, the Icon of the Target Executable will be copied to the Media file, every time the Media file is compiled.
This option is valid only if you selected a Target file and the file is EXE or DLL.

**Copy Icon From other Application/File**
Use this option to copy the icon from another EXE, DLL or ICO file.

# Execute Engine

In this page you will define operations to be performed during the Setup or UnInstall process. The operations can be Conditional or UnConditional. You may define sophisticated conditions with up to 3 rules for each condition. You can also select at what point of time, during Setup or UnInstall, each operation will be performed.

**IMPORTANT -** "Execute" is available only in the STUDIO and PRO Versions.

## Add Execute Item

To add Execution item click the **[Add Item]** button.
To read more click Add Execution Item.

Detailed description of all the Condition & Execution commands may be found at:
 www.pantaray.com/execute.html

**IMPORTANT RECOMMENDATION**
Whenever you enter a pathname, surround the entire path with quotes, this is especially important if your path include spaces. Whenever possible avoid using pathnames that include spaces.

## Test

After you have defined an Execution Item you can test it by clicking the **[Test32 !]** or **[Test64 !]** buttons.
The program will attempt to run the Execution Item and display a short log that describes the results of the test.
Please take care while testing because the program will attempt to really perform the operation.
Also you must take into account that not all conditions are available at the time of the test. For instance <Application Folder> is not yet created.
Use the **[Test32 !]** button to test execution items that should run in a 32-bit setup state.
Use the **[Test64 !]** button to test execution items that should run in a 64-bit setup state.

**Copy/Paste/Rename**
Use the **[Copy] [Paste]** & **[Rename]** buttons to create one or more duplicates of a certain Execution Item.

**Import/Export**
Use the **[Import]** & **[Export]** buttons to store and retreive a certain Execution Item to/from a disk file.
The extension of an exported execute item file is *.qspexec.

**Enable**
After you have defined an Execution Item You can disable its operation by UnChecking the Enable Checkbox. A disabled item will not be executed during Setup or UnInstall process.

**Directories**

The execution items are doing extensive use of different directories found on the target computer.
Detailed explanation about this subject may be found in "Directories".

## Service Files

Use **Service Files** when you need to supply certain files for the proper operation of a certain execution item.
The files you add to this list will be extracted to the TEMP directory with all the other files of the setup.
You may then refer to those files using the **<SrcDir>** directory.
When setup ends, all Service Files will be erased from the TEMP directory.

**Use For UnInstall**

Use this CheckBox to mark a file, If this file is needed for Execute Operation during UnInstall.
When a file is marked this way, the Setup program will copy this file from the TEMP directory to the Target Directory during Setup.

## Execution DLL File

We have included in the program several predefined **Conditions** and several predefined **Executions**.

However if you need a special Condition or Execution that is not included in the program, you can add it yourself by including it in a special purpose DLL.

To read more click Execution DLL File.

# Billboard

The billboard is a special window that is used to display images and text during the "Copy Files" stage of the installation. The billboard can accept a combination of any of the following type of files:

- BMP - Bitmap files.
- WMF- Windows Meta files.
- EMF - Enhanced Meta files.
- TXT - Ascii Text files.

When the Billboard is active, the normal dialog of the Setup program will disappear, and a special small copy progress bar will be displayed along with the billboard screen. The text and images you define will display on the billboard in a sequence controlled by a special timer.

Please note that you can mix text and images in one session.

## General

**Add Billboard**
You must Check this Checkbox for the Billboard to become operative.

**Add...**
Click the **[Add...]** button to add file(s) to the Billboard.

**Update...**
Click the **[Update...]** button to update a file in the Billboard.

**Delete...**
Click the **[Delete...]** button to delete files from the Billboard. You can also uncheck a file to eliminate it from the Billboard display.

**Preview**
Click the **[Preview]** button to see a test preview of the Billboard.

**Arrows**
Use the arrows to set the order of the files. You can also use Drag & Drop for the same purpose.

**Delay (Sec)**
Set the Billboard time interval. The time is counted in Seconds.

**Auto Size**
When **Auto Size** is Checked the Billboard will automatically adjust its size to the size of the current image.
Auto Size has no effect when text is displayed.

**3D Frame**
When **3D Frame** is Checked the Billboard will have a Classic Windows 3D frame around it.

**Show all**
When **Show All** is Checked the Billboard will display all its selected files even if the Copy operation has already terminated.

**Transparent**
When **Transparent** is Checked the Billboard will display all BMP files as transparent. The transparent color is always the first pixel on the bitmap. Please note that on most Bitmap the first pixel is the first on the bottom line, on some bitmaps the first pixel is the first on the top line.
This Checkbox has no effect on WMF, EMF and TXT files.

**Continues**
When **Continues** is Checked the Billboard will work continuously. When all presentations are shown and the Installation is not completed yet, the billboard will start displaying all the presentations from the first one.

**Enable Cancel Button**
Use this option to Enable/Disable the [Cancel] button on the Billboard progress bar panel.

**Font**
Click the **[Font]** button to define the following Font attributes: Name, Size, Style, Color and Script. This font is used for displaying text.

## Frame

**Width**
Set the Width of the Billboard in screen pixels. This value is ignored when displaying Images and "Auto Size" is checked.

**Height**
Set the Height of the Billboard in screen pixels. This value is ignored when displaying Images and "Auto Size" is checked.

**Margin**
Set the Margin that will surround the image and the text. This value is ignored when displaying Images and "Auto Size" is NOT checked.

**Location**
Click the **[Location]** Button to define the location of the Billboard.
Select any of the following locations:
Center, Top/Left, Top/Right, Bottom/Left, Bottom/Right, Top/Center, Bottom/Center, Left/Center, Right/Center.

Use the DX & DY values to shift the Billboard horizontally and vertically from the original location.

## Background Color

Here you will define the Background color of the Billboard.
The Background color will have effect in the following cases:
- When displaying text files.
- When displaying transparent WMF & EMF files.
- When the "Margin" value is set to a non zero value.

**Top Color & Bottom Color**
Use these color settings to define the color of the background.
The setup program will display a gradient that changes from the "Top Color" to the "Bottom Color".
If you prefer a solid color background, uncheck the "Bottom Color" Checkbox.

## Text Files

QSetup can easily display text files on the Billboard. The file must be a "pure ASCII text file" that may be edited with NOTEPAD. The font properties of the displayed text is defined as described under "Font".

**Text Headers**
Any line in the Text file that starts with the character ~ will be considered as a Header line and displayed in a size that is twice as large as the size of the rest of the text.

# Auto Update

## The Concept

"Auto Update" is a powerful concept. Using this option you will be able to Update your customers automatically from the Internet, or from a central file server - when ever you have a new version of your software available.

**Auto Inform**
"Auto Inform" will only Inform your user about a new update, and prompt him to download the new update from the internet using the browser.

**Auto Inform** is described in detail at the end of this document.

**IIS Server**
If you are hosting your website on an IIS server, you will have to set some new MIME types for "Auto Update" & "Auto Inform" to work properly. For more information read the following page: www.pantaray.com/iis.html.

IMPORTANT - "Auto Update" and "Auto Inform" are available only in the PRO version.

## How Does it Work?

When you add AutoUpdate option to your setup, QSetup will add a small Agent to the Setup delivery. This Agent is placed during setup in the "Application Folder".

The Agent is installed in such a way that it will run in parallel to your program.
The Agent will connect to the Internet from time to time and check if a new version is available.
If a new version is found, the Agent will download it using HTTP protocol, and place it in the Application directory.

When download is complete, the Agent will perform the following steps:
- Instruct your application to shut itself down (if needed).
- Extract all required files from the downloaded file.
- Run the update setup and replace old files with new ones.
- Restart your application (if needed).

IMPORTANT - Windows does not allow to overwrite a running executable (*.EXE, *.DLL, *.OCX, etc...).

## Terms in Use

- **Original Setup** - The basic setup you distribute to your customers, compiled with "QSetup Installation Suite".
- **Update Setup** - The collection of files that will be delivered to your customers during the Auto Update process.

## General

In this area you will enter data required for both the "Original Setup" and the "Update Setup".

**IMPORTANT -** You MUST fill all the data items in this area before you proceed to the other areas.

**Name of UPDATE Project:**
Enter here a file name without extension. This name will later on be used to define 4 different files. If you use the name "PhoneBookUpdate" then the following files will be produced:

- **PhoneBookUpdate.EXE** - the AutoUpdate Agent.

- **PhoneBookUpdate.ORIGINAL** - A small text file that will be added to the Original Setup. This file contains information required for the Auto Update procedure. This file will be created when you compile the Original Setup.

- **PhoneBookUpdate.INFO** - A small text file that will be added to the Update Setup, This file contains information required for the Auto Update procedure. This file will be created when you compile the Update Setup. This file will be downloaded from the Internet to check if new update is available.

- **PhoneBookUpdate.UPDATE** - This file will include the actual data of the update, and will also be downloaded from the Internet.

**Download URL of UPDATE Project**
This is the URL you will use to post the INFO file and the UPDATE file once a new update is available for download.
This must be an HTTP url.
    Example:  **http://www.microsoft.com/download**

- This URL can be the same one you are using to host your Original Setup.
- You can also specify here a normal directory path. This option is useful if you want to perform Auto Update from a central file server.

**Secret Token**
The secret token is a means for added security. It is encrypted and embedded in both the Original Setup and the Update Setup. Update will only take place if the "Secret Token" of both setups match.
Enter here any text you want.

**IMPORTANT** - for added security, the Secret Token is stored in your registry - not the QSP file.
Take care to store a copy of the Secret Token in a safe place where you can recover it, in case your registry is damaged.

**Add known extensions to files (IIS Server)**
The IIS Server will not allow by default to download files with the extensions ".INFO", ".UPDATE" & ".INFORM".
If you are placing your files on an IIS server, we recommend that you check this option.
This option will modify the extension:
- ".INFO" to ".INFO.TXT".
- ".UPDATE" to ".UPDATE.BIN".
- ".INFORM" to ".INFORM.TXT".

**IMPORTANT -** this option should be used only on newly created setups.

## Original Setup

To make your application "Auto Update" or "Auto Inform" enabled you must plan ahead. In this area you will enter the information required for your application to Auto Update itself form the Internet in the future.
All data you enter here will be stored in the Original Setup files.

**Auto Update Enable**
Check this CheckBox to make your application "Auto Update" or "Auto Inform" enabled.

**Probe for New Update**
Determine here how often the Agent will test for a new version.
To reduce Internet traffic we recommend that you use "Probe Frequency" that is not less then 1 hour.
If you select "Only at Agent Start" then the Agent will probe only once - just after it was started.
If you select "NEVER" then the Agent will NEVER probe automatically, this option is available for Advanced users. To read about Advanced options go to "Advanced Auto Update".

**Version of Original Setup**
Set here a number that will represent the current version of your Setup. We suggest that you start with **(1)**.
Only AutoUpdates that are marked with HIGHER number will be downloaded.
- We use here only PURE numbers (1, 2, 3 etc...) for clarity and dependability.

**ReAsk to Update after**
A user may decline when requested to confirm Auto Update.
If you set here a value greater then Zero, then the Agent will request for permission to update after the set days have passed from the previous attempt.
If you set here a value of Zero the user will be prompted to update only ONCE for each new update version.

Run Auto Update Agent at StartUp
If you Check this option then the Agent will start running as soon as your customer boot their PC.
Another option will be explained later.

**Add Auto Update Shortcut**
If you Check this option then your user will have the option to manually check for new update from the Start/Program menu.

**Set Shortcut Folder**
Using this option you can specify a sub-folder in the Start/Program menu to locate the shortcut.

**Set Shortcut Icon**
Using this option you can specify an icon the shortcut.

**Run As Administrator**
Since the original setup is always installed using admin right, Win7 and up - in some cases - may require this shortcut to be run with admin rights also.

We strongly reccomend that you keep this option always checked.

**Create Auto Update Log**
If you Check this option then the Agent will produce a log file that will record all agent activities.
This log will be located in the same directory where the Agent is located.
We recommend that you enable this option only when you debug your setup.

## Time to Update

Several weeks or months have passed. You already have a new version and you want to AutoUpdate all your customers.

There are 2 options for creating the Update file:
- Use the same Setup file that you use to distribute the new version (Single file or Split file).
- Create an UPDATE File that will be delivered only to your old customers.

In both cases make sure you increment the value at "Version of Original Setup", so that the new update becomes ORIGINAL for the next future update.

## Update Setup

When you prepare an Update delivery, you actually prepare 2 files. Assuming your Media file has the name "PhoneBookSetup" and the UPDATE Project name is "PhoneBookUpdate" then the following files will be created:

- "PhoneBookUpdate.INFO" - This is a small text file that will be downloaded first from the Internet by the Agent. Based on the information in this file, the Agent will decide if the new update is relevant for further processing.

- The second file is the file that includes the actual data required for the update. For description of this file read the next paragraph.

**Select UPDATE File**
In this ComboBox you will have the option to select one out of 3 files. Assuming your Media file has the name "PhoneBookSetup" and the UPDATE Project name is "PhoneBookUpdate" then you will be presented with the following options:
- PhoneBookUpdate.UPDATE
- PhoneBookSetup.EXE
- PhoneBookSetup.SPLIT

The first option is a special file that will be created when you click the **[Compile]** button.
The second option is the regular setup file you created to distribute your new version, compiled as SINGLE SelfExtract file.
The third option is the regular setup file you created to distribute your new version, compiled as SPLIT setup.

Running Process
Enter here the name of the program that must be STOPPED before the actual update begins. Normally you will enter here the name of the Target Executable.
If there is no need to shut down any program, leave this text empty.

**Shut Down Method**
Specify here how to Shut Down the Running Process.
The following options are available:
- Don't shut down.

- Wait for process to close on its own - This is good for programs that are started and stopped several times every day.
- Issue a Message Box - In this case the Agent will issue a message box requesting the user to shut down the application before Auto Update starts.
- Using WM_CLOSE - the Agent will send WM_CLOSE message to the application, ordering it to orderly shut itself down, and probably giving the user the chance to save all open files before shut down.
- Using WM_QUIT - the Agent will send WM_QUIT message to the application, ordering it to shut itself down, probably giving the user NO chance to save all open files before shut down.
- Using Terminate Process - This is the brute force option, This is the method Windows is using to force a process to shut itself down.

**Request Confirmation before Download**
If this option is Checked, a message box will appear requesting the user to confirm the download.
If **Force** is checked the user will not have the option to abort the process.

**Request Confirmation before Install**
If this option is Checked, a message box will appear requesting the user to confirm the installation.
If **Force** is checked the user will not have the option to abort the process.

**Perform Background Update**
If this option is Checked, the whole update process will be performed in the background.

**Inform User when Update is Finished**
If this option is Checked, then once the update process is finished the Agent will display a message.

**Restart Process when Update is Finished**
If this option is checked, the Agent will restart the Running Process at the end of the Update process. Restart will take place only if the Agent has actually shut down the process previously.

**Version of New Update**
Enter here a number that will represent the version of the new release. The number you enter here MUST be greater then the number you specified in "**Version of Original Setup"** when you created the Original Setup.

- We use here only PURE numbers (1, 2, 3 etc...) for clarity and dependability.

**Valid for Versions:**
Specify here all the old versions that the current delivery can AutoUpdate.

   **Example:**
   - Your current version is 4.
   - You already released Versions 1,2 & 3 to the public.
   - The current version can only safely update versions: 2 & 3.
   - In this case enter **2,3** in this field.

If you leave this field empty, then ALL previous versions will be updated.

<div align="right">Test Mode</div>

When you create an Auto Update delivery, you obviously want to test it before you send it to your customers. To test it you will have to run a copy of the older version on your PC, and then upload the update delivery to the internet. the problem with this scenario is that while you are still testing, your customers will also download the new update and install it even before you conclude the test.

To avoid this scenario, check the "Test Mode" CheckBox.

When "Test Mode" is checked any Agent running on your computer will look for the .INFO file in the "Project Directory" on your local HardDisk. This way you can test the new setup without Uploading it to the internet.

**Compile**

Click this button to create the INFO file. The name of this file will be the one stated under "Name of UPDATE Project". If you selected to deliver the data in a .UPDATE file then this file will also be compiled.

**Upload**

Click this button to UPLOAD the .INFO file and the selected data file to the web using FTP protocol.

**Verify URL**

Use this dialog to verify that the URLs you specified for the INFO & DATA files are correct.

This dialog will only test INTERNET URLs.

Please remember to test the URLs even before you release the original version of your software.

## Running the AutoUpdate Agent

As explained above, the AutoUpdate Agent must be set to run on your customers PC for the AutoUpdate process to function.

As explained under **Run AutoUpdate at StartUp,** you can instruct the setup program to make the agent run at Windows startup. Another way would be to start it from your program using the WinExec command under Windows.

<div align="right">Example:</div>

If the "Name of UPDATE Project" is "PhoneBookUpdate" then the Agent name is "PhoneBookUpdate.EXE". Add to your program a line similar to the following:

```
WinExec("PhoneBookUpdate.EXE",0);
```

If the Agent is located in a parent directory relative to your application, use a line similar to the following:

```
WinExec("..\PhoneBookUpdate.EXE",0);
```

If the Agent is located in a child directory by the name XXX use a line similar to the following:

```
WinExec("XXX\PhoneBookUpdate.EXE",0);
```

You can also STOP the agent from inside your program using a line similar to the following:

```
WinExec("PhoneBookUpdate.EXE /STOP=0",0);
```

if you use a line similar to the following, then the Agent will stop 300 seconds AFTER issuing the command:

```
WinExec("PhoneBookUpdate.EXE /STOP=300",0);
```

if you use a line similar to the following, then the Agent will perform a test in the background and exit immediately after the test.

```
WinExec("PhoneBookUpdate.EXE /TEST=1",0);
```

if you use a line similar to the following, then the Agent will perform a test in the foreground - displaying various dialogs during the process - and exit immediately after the test.

```
WinExec("PhoneBookUpdate.EXE /TEST=2",0);
```

# Advanced Auto Update

QSetup includes a comprehensive API that will let you control the Auto Update Agent from within your Running Application. To read about this option go to "Advanced Auto Update".

# Auto Inform

"Auto Inform" is quite similar to "Auto Update".
The first 2 steps are identical. Just like in "Auto Update" you must fill the first 2 areas "General" & "Original Setup" as explained above.

When you have a new version available you must prepare the following:
1. Create a stand alone setup as an EXE file and upload it to your website.
2. Prepare a special download page on your website with a link to the new setup executable.
3. Prepare an RTF or TXT file that will describe the new update.

On the "Auto Update" area select the "Auto Inform" tab and do the following:
1. Check the "Select INFORM File:" option.
2. Click the [...] button and select the RTF or TXT file you prepared.
3. Set the size of the message your user will see.
4. Check the "Download URL:" option.
5. Enter the full URL to the download page you prepared.

**Preview**
Now Click the [Preview...] button.
QSetup will display a special dialog displaying the RTF or TXT file you created.

Click the [Download...] button on the dialog and the browser will open with the page you indicated as the "Download URL".

**Version of New Update**
Set this value as described above for "Auto Update".

**Valid for Versions:**
Set this value as described above for "Auto Update".

**Compile**
Click this button to create the INFO & INFORM files. The names of the files will be the one stated under "Name of UPDATE Project".

Upload
Click this button to UPLOAD the .INFO & .INFORM files to the web using FTP protocol.

**Verify URL**
Use this dialog to verify that the URLs you specified for the INFO & INFORM files are correct.
This dialog will only test INTERNET URLs.
Please remember to test the URLs even before you release the original version of your software.

# Merge Modules

A "Merge Module" (.msm file) is a single package that includes all files, resources, registry entries, and setup logic to install a shared component. A Merge Module is intended to be embedded in an MSI file.

MSI stands for "Microsoft Installer" (Which was later renamed by Microsoft to "Windows Installer").
MSI is a new Installation technology developed by Microsoft.
MSI is available by default in the following Windows versions: WinME, Win2000, WinXP, Win2003, Vista, Win7, Win8 & Win10.
It is also available as a free download from the Microsoft website for earlier versions of Windows.

Microsoft and other software companies are now distributing shared software components as "Merge Modules"

If you want to distribute a "Merge Module" to your customers as part of your setup file, the "Merge Module" must be wrapped in an MSI file when the file.

IMPORTANT - The "Merge Modules" option is available only in the PRO version.

## Add Merge Module

To add a "Merge Module" file to your setup click [Add Item...] and select the MSM file you want (you may add more then one file).

## Options

### Install At
The MSI file will be installed during the "Copy Files" stage of the setup. You may select one of two options:
- Copy Start - Just before the first file is copied to the target directory.
- Copy end - Just after the last file is copied to the target directory.

### UnInstall At
You may select one of three options for the UnInstallation of the Merge Module(s).
- Ignore - Don't UnInstall the Merge Module(s).
- UnInstall Start - The first item to be uninstalled from the PC.
- UnInstall End - The last item to be uninstalled from the PC.

### Display Options
- **Hidden** - Don't show any MSI dialog during Setup & UnInstall.
- **Passive** - Show a limited MSI Dialog during Setup & UnInstall (progress bar only).
- **Full** - Show the full MSI dialog during Setup & UnInstall

### Always Recreate MSI File
When this option is checked the MSI file will be recreated every time you compile the setup.

### MSI File Name

By default QSetup will offer a name for the MSI file, however you may modify the name as you wish.

After you have selected all the options according to your preferences, click the [Compile] button.
QSetup will create the MSI file (if needed) and proceed to compile the setup file as usual.
During compilation QSetup will embed the MSI file in the setup and later on during setup QSetup will run the MSI file on the target PC.

## Set Parameters

Some Merge Modules introduce parameters that you can set before you compile your setup.
Highlight the the Merge Module name and click the [Set Parameters...] button.
In the dialog that opens scroll through the list of parameters and set every parameter as required

## Test

**Create MSI File**
Click this button to create the MSI file. The file will be stored in the Project directory where the Setup file is normally created.

**Install MSI File**
Click this button to test the newly created MSI file. When you click this button, the Merge Modules you are using will be actually installed on your PC.

**UnInstall MSI File**
Click this button to UnInstall the MSI file you just install. It is always recommended that after you install the MSI file for testing, you will also UnInstall it. (Normally an MSI file can not be installed twice without first be UnInstalled).

When the **Install** & **UnInstall** operation is complete, a special "Result Message" will appear with the operation result.
For "MSI Error Codes" click the following link:
www.pantaray.com/msi_error_codes.html

# UnInstall

## Create UnInstall

Create Automatic UnInstall

If this option is checked then QSetup will create an UnInstall option for the program you are installing. Normally the user will call this option from the "Add/Remove Programs" dialog of the "Control Panel".
You may also instruct the program to add an UnInstall shortcut to the Start/Program menu, by checking the "Add UnInstall Shortcut" Checkbox found on the "Shortcut" Checkbox found on the "Shortcut" page.

**Scramble Data File**
When the Setup program is running it creates a data file that is later used for the UnInstall program. If this option is checked, the file will be scrambled.

**Hide Error Message at end of UnInstall**
If Errors occurs during the UnInstall process, QSetup will display an error message. Check this option if you want to hide this Error Message.

**Hide Log Button at end of UnInstall**
If Errors occurs during the UnInstall process, QSetup will display Log Button on the UnInstall dialog. Check this option if you want to hide this Log Button.

**Perform Silent UnInstall**
When this option is checked, the UnInstall program will run without intervention of the user.
You can achieve the same effect by adding **/silent** parameter on the command-line of the UnInstall program.

Perform Hidden UnInstall

When this option is checked, the UnInstall program will run completely in the background, presenting no dialog during the process.
You can achieve the same effect by adding **/hide** parameter on the command-line of the UnInstall program.

UnInstall Exe Stamp

When adding UnInstall option, QSetup add a file by the name un_?????_NNNNN.exe to the setup delivery.
- This file is actually the UnInstallation program.
- The ????? part is actually the "Media File Name" as defined on the "Project" page.
- The NNNNN part of the name is calculated from the target executable file name.
- Using this system you can install several programs into one directory and each one will have its own UnInstall program.

The "UnInstall Exe Name" option will let you specify manually a specific name.

Set UnInstall Exe Dir

By default the UnInstall files (EXE & TXT) will be placed in the <Application Folder> during setup.
Use this option to set another location for the UnInstall files.

## UnInstall Shortcut

Add UnInstall Shortcut

If you check this Checkbox, the setup program will add an "UnInstallation Shortcut" to the Start/Program menu. For this item to be active, you must check the "Create Automatic UnInstall" Checkbox in the "Switches" page.

**Set Shortcut Sub Folder**
Using this option you can specify a sub-folder in the Start/Program menu to locate the Shortcut.

**Force Remove Main Shortcut Folder**
Normally the UnInstall program will attempt to remove the main shortcut folder.
If the user has added manually some more shortcuts to this Folder, or another installation program has placed shortcuts in the same Folder, the Folder will NOT be removed during UnInstall.
You can FORCE the UnInstall program to remove this folder by checking this option.

## Add/Remove Programs

Add Shortcut to "Add/Remove Programs"

Check this option to add an UnInstall shortcut to the "Add/Remove Programs" dialog.

Add "Change" Button

Check this option to add a **Change** button to the UnInstall entry in the "Add/Remove Programs" dialog.

Display Name

By default QSetup will use the "Program Descriptive Name" as the Display Name of this shortcut.
Use this option to specify a different name.

Display Icon

Use this option to set a specific Icon for this shortcut.

**Support Information URL**
Add here a URL that will lead to the support information page on your website.

**Product Updates URL**
Add here a URL that will lead to the product updates page on your website.

# Directories

In several places in the program ("Files" page, "Registry" page, "IniFile" page, "Environment" page, "Execute" page, etc...) you will find a ComboBox with a list of aliases for some predefined directories. The ComboBox is accompanied by an adjacent Editbox were you should enter the rest of the directory definition.

The directories should be interpreted as follows:

| | |
|---|---|
| **<AbsoluteDir>** | Refer to the definition in the Editbox as an absolute directory. |
| **<InstallDir>** | The target installation directory as defined under "Application Folder" in the "Files" page of the Composer. If the user selects a different destination directory, the content of this alias will change accordingly. |
| **<InstallCommonDir>** | The target installation directory as defined under "Common Folder" in the "Files" page of the Composer. |
| **<InstallAuxDir>** | The target installation directory as defined under "Auxiliary Folder" in the "Files" page of the Composer. |
| **<WinDir>** | The Windows directory - normally "C:\Windows". |
| **<WinSys32Dir>** | The System Directory - normally "C:\Windows\System32". |
| **<WinSys16Dir>** | The OLD System Directory - normally "C:\Windows\System". |
| **<ProgramFilesDir>** | The "Program Files" directory as indicated in the Registry - normally "C:\Program Files". |
| **<CommonFilesDir>** | The "Common Files" directory as indicated in the Registry - normally "C:\Program Files\Common Files". |
| **<FontDir>** | The directory where Windows stores font files - normally "C:\Windows\Fonts". |
| **<SrcDir>** | The directory where the Setup or UnInstall program is now running from. |
| **<TempDir>** | The Windows TEMP directory - normally "C:\Windows\TEMP". |
| **<UserDir>** | The User Home directory - normally "C:\Documents and Settings\UserName". |
| **<OriginDir>** | The directory where the original setup file was running from - normally it will be the CD-ROM. |
| **<ServiceDir>** | The directory where Service Files for the "Execute Engine" will be found. |
| **<InstallDrive>** | The Drive of the target installation directory <InstallDir> - normally "C:\". |
| **<WinDrive>** | The Drive where Windows is installed - normally "C:\". |
| **<WinSysDrive>** | The Drive where the System directory is located - normally "C:\". |
| **<OriginDrive>** | The Drive where the original setup file was running from - normally it will be the CD-ROM drive (E:\). |
| **<InstallDisk>** | The Disk of the target installation directory <InstallDir> - normally "C:". |
| **<WinDisk>** | The Disk where Windows is installed - normally "C:". |
| **<WinSysDisk>** | The Disk where the System directory is located - normally "C:". |
| **<OriginDisk>** | The Disk where the original setup file was running from - normally it will be the CD-ROM drive (E:). |

<UserAppDataDir>
<UserLocalAppDataDir>
<MyDocumentsDir>
<UserStartMenuDir>
<UserStartProgramMenuDir>
<UserDesktopDir>

<AllUsersDir>
<AllUsersAppDataDir>
<AllUsersDocumentsDir>
<AllUsersStartMenuDir>
<AllUsersStartProgramMenuDir>
<AllUsersDesktopDir>

<DocumentsAndSettingsDir>

**<RunTimeDir_1>**          The directory will be located on the target PC at runtime, based on its name or content.

**<RunTimeDir_2>**          The directory will be located on the target PC at runtime, based on its name or content.

   ...

**<RunTimeDir_8>**          The directory will be located on the target PC at runtime, based on its name or content.


## <RunTimeDir_#>

Using <RunTimeDir_#> you can instruct QSetup to locate a specific directory at runtime (that is - when the setup is running on your customer's PC) - based on files found in that directory or the name of the directory or some Registry information.

Upto 8 RunTime directories may be defined with the names: **<RunTimeDir_1>, <RunTimeDir_2> ... <RunTimeDir_8>**.
To define any of those directories click the corresponding button on the **Files** page.

Once you define a RunTime directory, you can use it in the **Files** page as well as in the **Execute** & **IniFile** pages.

QSetup will attempt to find the RunTime directory using 4 methods:

1. **Search for Application Directory in the Registry**
   When this method is selected, QSetup will attempt to read the directory from the following key in the Registry:
   HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\App Paths\ProgExe
   This method is very effective for BIG applications like WinWord.exe & Excel.exe.
   When selecting this option enter the executable file name with extension as parameter:
   Sample: **winword.exe**, **excel.exe**, etc...
   This search is very fast.

2. **Read Directory name from the Registry**
   When this method is selected, QSetup will attempt to read the directory directly from the Registry.

Use this method to read a value that was entered by a previous copy of your program or by another program.
Enter here a **KeyName** and read its **default** value, or Enter here a **KeyName\ValueName** combination and read its value.
This search is very fast.

### 3. Search for Directory of Specific File
When this method is selected, QSetup will attempt to find the directory using a normal FindFirst/FindNext search pattern.
When selecting this option enter the file name with extension as parameter:
This search is very slow.

### 4. Search for FullPath of Specific Directory
When this method is selected, QSetup will attempt to find the directory using a normal FindFirst/FindNext search pattern.
When selecting this option enter the directory name. For added security you may add the path of the parent directory.
For example:
when searching for **Office** the search may find:
    "C:\Program Files\Microsoft Office".
when searching for **Microsoft\Office** the search may find:
    "D:\Documents and Settings\Guest\Application Data\Microsoft\Office".
This search is quite slow.

You can use any mixture of the above mentioned searches.
The search will be executed in the order of the dialog.
Once a suitable directory is found, the search will stop.
You may also define a default directory that will become effective if no suitable directory is found.

### Show Message
Scanning the drive for a specified directory is usually a lengthy operation.
Check this option to display the message "Searching for Directory --- Please Wait!" during the search.

### If Directory not Found
If the <RunTimeDir_#> is not found you can instruct the Setup program to issue an error message and/or abort the installation.
The actual test for the existence of the <RunTimeDir_#> is performed just before the directory is needed for the first time.

# Aliases

Where ever it is legal to enter a directory alias you may also enter any of the following tokens:

**<UserName>**
This Token will be replaced at runtime with the current user name (normally the name the user used at windows installation).

**<LoggedUserName>**
This Token will be replaced at runtime with the current user name (normally the name the user used at logon).

**<CompanyName>**
This Token will be replaced at runtime with the current company name (normally the name the user used at windows installation).

**<ComputerName>**
This Token will be replaced at runtime with the current computer name (the name of the computer  the user is running on now).

**<CurrentDomainName>**
This Token will be replaced at runtime with the current domain name (the domain the user used at logon).

**<EnteredUserName>**
This Token will be replaced at runtime with the User Name as entered by the user in "User Information" dialog.

**<EnteredCompanyName>**
This Token will be replaced at runtime with the Company Name as entered by the user in "User Information" dialog.

**<EnteredSerialNum>**
This Token will be replaced at runtime with the Serial Number as entered by the user in "User Information" dialog.

**<Date=#>**
Use this token to enter the current date.
This token should have any of the following shapes:

| | |
|---|---|
| **<Date=0>** | The number of days that have elapsed since 01-JAN-2000. |
| **<Date=1>** | 28-02-07 |
| **<Date=2>** | 02-28-07 |
| **<Date=3>** | 07-02-28 |
| **<Date=4>** | 28-02-2007 |
| **<Date=5>** | 02-28-2007 |
| **<Date=6>** | 2007-02-28 |
| **<Date=7>** | 2007-02-28 12:31:59 |

**<VendorCompanyName>**
This Token will be replaced at runtime with the name of the company that produced the setup - as entered on the "Project" page.

**<ProjectName>**
This Token will be replaced at runtime with the "Project Name" - as entered on the "Project" page.

**<ProgramDescriptiveName>**
This Token will be replaced at runtime with the "Program Descriptive Name"  - as entered on the "Project" page.

**<ProgramVersion>**
This Token will be replaced at runtime with the "ProgramVersion"  - as entered on the "Project" page.

**<UnInstallExePath>**
This Token will be replaced at runtime with the full path of the UnInstall executable.

**<LastExecutableExitCode>**
This Token will be replaced at runtime with the Exit Code of the last "Run Executable" command.

# MSI

MSI is an Installation technology developed by Microsoft.
MSI stands for "Microsoft Installer" (Which was later renamed by Microsoft to "Windows Installer").
MSI is available by default in the following Windows versions: WinME, Win2000, WinXP Win2003, Vista, Win7, Win8 & Win10.
It is also available as a free download from the Microsoft website for earlier versions of Windows.

Starting from version 9.0 QSetup is able to produce setup files in the form of an MSI file. Once you define a setup using the Composer you can produce both a classic "Self Extract" setup in the form of an EXE file or an MSI file. To select EXE or MSI use the ComboBox located at the **Bottom/Right** side of the Composer screen.

## Why MSI?

MSI is mainly required by IT professionals in large organizations. For the IT professional the MSI format offers the following benefits:

- Self healing - If a program was damaged on the end-user's desktop, the user can rerun the original MSI setup file to initiate an automatic repair process.
- MSI files can be distributed to many desktops in an organization using "Active Directory".
- MSI file can be installed by a user who does not have administrative privileges.
- An IT professional can modify an MSI file and adjust it to his organization's requirements before distribution to end users.
- The MSI Format supports a rollback function which will restore your PC to its original state if the setup process was interrupted.

## Producing an MSI file

To produce an MSI file using QSetup use the following procedure.

- Define a setup using some or all of the pages of the composer.
- Goto the Bottom/Left of the Composer screen and select "MSI" in the selection box.
- Click the [Compile] button.

As a result of this sequence of operations you will have a distribution file (Media File) with an MSI extension in the "Project Directory".

**MSI Log**
If you click the [Run] button and the "Debug" option is checked on the "Project" page, QSetup will display the MSI log after running the MSI setup file.

**IMPORTANT** - using the same setup definitions you can easily produce a classic "Self Extract" setup in the form of an EXE file. Just select "EXE" in the selection box and click [Compile].

When in MSI mode, clicking the [Run] button will launch the MSI file you just compiled.

When in MSI mode you can upload the MSI file you just compiled to the web by clicking the [Upload] button.

**IMPORTANT** - If you want to reinstall an MSI setup you must first uninstall it. You can uninstall the old setup from the Windows "Add/Remove programs" dialog.

## MSI vs EXE

Unfortunately not all of the features that are currently supported by the QSetup EXE format are available in the MSI format. We hope to add more functionality to our MSI implementation in future versions, however some features can not be implemented since the MSI format will not allow them.

## Missing Features

The following list summarizes the features the are not available in QSetup MSI implementation:

**Project Page**
- Language Support
- Compression Level
- Create Split Setup
- Span CDs
- Debug

**Display Page**
- Display page is not implemented

**Files Page**
- Exclusivity Tag
- Include Group by Setup Number
- Overwrite Files & Remove Directories are handled according to MSI policies.

**Dialogs Page**
- Add Image
- Show Progress Bar #2
- Perform Silent & Hidden Setup
- CD Setup
- Show Compact Setup Also
- Force Partial Setup
- Custom Dialogs

**Switches Page**
- Create Setup.log File
- Request Confirmation Before Extract
- Previous Installation (Handled by MSI).
- Test for running executable.
- Run/Run Once.
- Auto Run Test.

**Shortcuts Page**

- Language Support.
- Selected by End User.

**Registry Page**
- Install Reg Files.

**Properties Page**
- Properties Page is not implemented

**Execute Page**
- Execute Engine is not implemented

**Billboard Page**
- Billboard in not implemented

**Auto Update**
- Auto Update is not implemented

**UnInstall Page**
- UnInstall is handled automatically by MSI, as a result only few of the parameters in the uninstall page are valid also for MSI.

# Appendixes -

## Add/Update Execution Item

In this dialog you will define operations to be performed during the Setup or UnInstall process.

Detailed description of all the Condition & Execution commands may be found at:
   www.pantaray.com/execute.html

### Item Name:

Every execution item must have a name. Enter here a name that will describe the function of the execution item.

### Perform At:

Define here at what stage of the process the operation will be performed by selecting any of the following options:

- Undefined Stage
- Setup Start
- Copy Start
- Copy End
- Setup End
- UnInstall Start
- UnInstall End

- Before User Information
- After User Information
- Before Setup Type
- After Setup Type
- Before Custom
- After Custom
- Before Destination
- After Destination
- Before Associate
- After Associate
- Before Shortcuts
- After Shortcuts
- Before Complete
- After Complete
- After Billboard
- Before Dialogs
- After Reg Components

## Item Type:

Select from 3 different item types:
- Conditional  - The operation will be performed only if certain condition is meet.
- UnConditional  - The operation will be performed unconditionally.
- While Loop  - The operation will be performed repeatedly as long as the condition is meet - AND THEN optionally perform another operation.

### Loops:
This option is valid only if you selected "While Loop".
If you enter here a Zero then the While Loop will perform as long as the condition is meet (even forever).
Any number greater then zero means - the loop will perform upto this number of loops and then exit the loop even if the condition is not meet.

### Delay (MS):
This option is valid only if you selected "While Loop".
Selecting any number greater then zero will add a delay (in MilliSeconds) between every loop iteration.

## Online Help

Use this checkbox to display an online help that will describe every Condition or Execution item while you hover the mouse over this item.

## Conditions

We provide the following predefined conditions:

### Files & Directories
- File Found
- Directory Found
- Drive Found
- Text Found in File
- File Found (HTTP)
- File Attribute is in
- File is Locked

### Application
- Application Found
- Application Path Found
- Application Version is
- Application is Running

### Miscellaneous 1
- Exe is Running
- Service is Installed
- Service is Running
- Operating System is in
- File Version is
- HWindow is Found - (Using the FindWindow() API call)
- Local Language is in
- Selected Language is in
- File Size is (Bytes)
- File Date and Time is
- Internet Connection OK

- Last Executable Exit Code is
- Printer Intalled

**Registry**
- Registry Key Found
- Registry Value Found
- Registry Value is

**Environment**
- Environment Variable Found
- Environment Variable Is
- Environment Variable Is (UpCase) - (Ignore case when performing the check).

**Hardware**
- Memory Size is (MB)
- CPU Speed is (MHz)
- Screen Resolution is
- Color Depth is
- Disk Free Space is (MB)

**X64**
- Running on 64 Bit OS
- 64 Bit State is ON

**Modules**
- .NET Framework Version is in
- Visual Basic Runtime Found
- ADO(MDAC) Version is
- MS-ACCESS Version is
- SQL(MSDE) Version is
- ODBC Installed
- BDE Installed
- DirectX Version is
- Media Player Version is
- JDK Version is
- JRE Version is
- DAO Installed
- Visual J# Version is
- Internet Explorer Version is
- Flash Player Version is
- Acrobat Reader Version is
- SQL Express Version is
- MS JET 3.5 Version is
- MS JET 4.0 Version is
- IIS Version is
- Visual C++ Redistributable Found

**Miscellaneous 2**
- Ask Yes/No
- Ask OK/Cancel
- Setup Type is
- Group Selected
- Auto Update is Running
- Billboard is Running

**Admin**
- User Name is - (As entered in the User Information dialog).
- Company Name is - (As entered in the User Information dialog).
- Serial Number is - (As entered in the User Information dialog).
- User is ADMINISTRATOR
- User Privilege is in

**Custom Dialogs**
- Control Text is
- Control Text is Empty
- Control Item Index is
- Control is Checked

**Variables**
- Compare 1 Variable
- Compare 2 Variables
- Text Found in Variable
- Variable Found in Text
- Variable Found in Variable
- Variable is Empty
- Variable Length is

Most of the conditions accept one argument, some accept 2 or 3 arguments.
After you select a condition, click the Browse button next to Argument-1 or Argument-2 to select from the available arguments.

**NOTE**
**Application -** refers to all exe files registered in the registry under the following key:
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\App Paths
**Service** -       refers to special programs that are installed on NT class machines and are automatically started during the startup process of the computer.

## Stop/And/Or/Xor

If you want to combine 2 or 3 conditions you must define the relation between them.
If you are using 3 conditions, the first two will evaluate first and the result will be evaluated against the third condition.
Result=((**Condition-1** And/Or/Xor **Condition-2**) And/Or/Xor **Condition-3**)

You can negate any condition by checking the **Not** Checkbox that precede it.
If you want to use only one condition Click **Stop** after the first condition.
If you want to use two conditions Click **Stop** after the second condition.

**Partial Check**
When you combine 2 or 3 conditions, ALL conditions are evaluated first and ONLY then are related to each other.
By Checking **Partial Check** you can avoid some evaluations when not needed - for instance:
If you have 2 conditions with an **OR** relation and the First condition evaluated to TRUE, there is no need to perform the second condition because no matter what will be the outcome of this condition the end result will be TRUE.
If **Partial Check** is Checked - only the first condition will be evaluated.
If **Partial Check** is UnChecked - both conditions will be evaluated.

# Executions

We provide the following predefined executions:

**Files**
- Delete File(s)
- Copy File(s)
- Move File(s)
- Rename File(s)
- Set File(s) Attributes
- Allow Full Access File(s)
- Download File (HTTP)
- Open CAB File
- Copy File Delayed
- Get Short PathName
- Get Expanded PathName
- Get Network PathName

**Directories**
- Create Directory
- Remove Directory - (will be removed only if empty).
- Force Remove Directory - (first will erase all the files and then remove the directory).
- Set Working Directory
- Copy Directory Tree
- Rename Directory
- Set Folder to
- Set Setup Type
- Allow Full Access Directory
- Set Directory Attributes

**Application**
- Delete Application Executable
- Remove Application Directory
- UnInstall application
- Run Application
- Run Application and Wait - (Wait until the process closes)
- Shut Down Application
- Wait while Application is Running

**Executable**
- Shut Down Executable
- Run Executable
- Run Executable and Wait - (Wait until the process closes)
- Shell Execute
- Shell Execute and Wait - (Wait until the process closes)
- Run Batch File
- Run Batch File and Wait - (Wait until the process closes)
- Run MSI File
- Run MSI File and Wait - (Wait until the process closes)
- Delete Running Executable
- Wait while Executable is Running
- Wait while Shell App is Running
- Set Compatibility Mode

**NT Service**
- Create Service

- Create Interactive Service
- Create Kernel Service
- Create File Service
- Create Shared Service
- Create Interactive Shared Service
- Start Service
- Stop Service
- Pause Service
- Continue Service
- Remove Service
- Set Service Description

**Servers & Drivers**
- Register OCX/COM Server
- UnRegister OCX/COM Server
- ODBC Config Data Source
- Install Screen Saver
- Install .INF File
- Register .NET Assembly
- UnRegister .NET Assembly

**IIS**
- Create IIS Virtual Directory
- Remove IIS Virtual Directory
- Create IIS Application
- Remove IIS Application

**GAC**
- Install Assembly in the GAC
- UnInstall Assembly from the GAC

**X64**
- Set 64 bit State
- Set 32 bit State
- Restore Original Setup State

**Registry**
- Create Registry Key
- Remove Registry Key
- Create Registry Value (String)
- Create Registry Value (Integer)
- Create Registry Value (Hex)
- Create Registry Value (MultiString)
- Create Registry Value (ExpandString)
- Append Registry Value (String)
- Remove Registry Value
- Read Registry Value
- Append Registry Value (MultiString)
- Install REG File

**XML File**
- Write XML Value
- Delete XML Value
- Read XML Value
- Create XML Node

- Delete XML Node

**INI File**
- Write INI Item
- Delete INI Item
- Delete INI Section
- Read INI Item

**Environment**
- Create Environment Variable
- Remove Environment Variable
- Read Environment Variable

**Text File**
- Append Text to a File - (Add <P> for NewLine & <T> for TAB)
- Replace Text in File
- Replace Text in File CR+TAB

**File Association**
- Create File Association
- Remove File Association
- Set File Association Description
- Set File Association Icon

**Setup Dialog**
- Set Space Required on Drive
- ADD to Space Required on Drive
- Set Propose to Restart the PC
- Set Propose to Launch the App.
- Set Dialog to
- Set Group
- Refresh RunTimeDir
- Browse for Folder
- Browse for File
- Set Setup Type
- Enable Setup Dialog
- Enable Dialog Button
- Set Edit Text

**Custom Dialogs**
- Set Control Property
- Get Control Property

**Variables**
- Set Variable Value
- Perform Variable Math - ( + - * / )
- Replace Text in Variable
- Select one of many
- Input text string
- Input text string (many)
- Set Case Sensitive
- Set Whole Word

**Miscellaneous**
- Display Message - (Add <P> for NewLine & <T> for TAB)
- Show PopUp Message
- Hide PopUp Message

- Wait (MS)
- Set Setup Exit Code
- Restart the PC
- Exit - (Abort the setup)
- Break - (Stop execution of "Execute" item)

## THEN

If a condition evaluates to TRUE or the operation is unconditional, up to 3 operations will be performed. Define the required operation and make sure you check the Checkbox that precede it.

**IMPORTANT**:
You can have up to 6 THEN operations by UnChecking the **ELSE** CheckBox.

## ELSE

If a condition evaluates to FALSE, up to 3 operations will be performed. Define the required operation and make sure you check the Checkbox that precede it.

## Next & Prev

Use these buttons to scroll through all the execution items your project includes.

## Copy

Click this button to copy the content of the current Execution Item to the clipboard.

## Paste

Click this button to paste the content of clipboard to the current Execution Item.

# VARIABLES

Using the Execute Engine you can Set, Compare & Use internal variables.

The following commands are Variable related:
- Set Variable Value
- Select one of many
- Perform Variable Math
- Input text string
- Set Case Sensitive
- Set Whole Word
- Read Registry Value - (Into a variable)
- Read INI Item - (Into a variable)
- Read Environment Variable - (Into a variable)
- Compare 1 Variable
- Compare 2 Variables
- Text Found in Variable
- Variable Found in Text
- Variable Found in Variable
- Perform Variable Math

Detailed description of the commands may be found at:
www.pantaray.com/execute.html

A variable is a string item in the form VariableName=VariableValue.
Where ever it is legal to enter a directory alias you may also enter a VariableName in the
form: <VariableName>.
Variables are kept across the entire Setup session, thus you can define a variable at
"Setup Start" and use it a "Setup End".
Variables are also saved from setup to uninstall, thus you can define a variable during the
setup process and use it during the UnInstall process.

## Variable Examples:

Command: **Set Variable Value**
Argument-1:    MyDirectory
Argument-2:    C:\Program Files\MySampleDir

- This command will create the following variable:
   Name: "MyDirectory"
   Value: "C:\Program Files\MySampleDir"

Command: **Display Message**
Argument-1:    My Directory Name is: <MyDirectory>

- When run this command will display a message box with the text:
   My Directory Name is: C:\Program Files\MySampleDir

Command: **Create Directory**
Argument-1:    <MyDirectory>

- When run this command will create the following directory:
   C:\Program Files\MySampleDir

Command: **Set Variable Value**
Argument-1:    MyDirectory2
Argument-2:    MyPersonalDir

- This command will create the following variable:
   Name: "MyDirectory2"
   Value: "MyPersonalDir"

Command: **Create Directory**
Argument-1:    <WinDir>\<MyDirectory2>

- When run this command will create the following directory:
   C:\Windows\MyPersonalDir

## COMPARING Variables

The following 2 condition commands maybe used to compare variables:
- Compare 1 Variable
- Compare 2 Variables

The first command compare a variable value with a constant.
The second command compares 2 variables.

As mentioned before all variable values are kept as strings - however before a compare is performed, QSetup will attempt to convert both items to a numerical value (Integer or real), only If BOTH attempts are successful the comparison will be made on numerical values otherwise it will be a string compare.

**EXAMPLE:**
   11.9 is greater then 9.11 - (Numerical compare).
   11_9 is smaller then 9_11 - (String compare and "1" is smaller then "9").
**Set Case Sensitive**

Use this command to define if String Compares and String Searches are **CaseSensitive**.
By default **CaseSensitive** is TRUE.

**Set Whole Word**
Use this command to define if String Searches are performed on **WholeWords** only.
By default **WholeWord** is FALSE.

## Command Line Parameters

When running the setup program you can define variables on the command line to be used by the "Execute Engine".
The syntax is simmilar to the following:

/[Var1]=1234 /[MyName]=John

There is no limit to the number of variables you can define in this way.

# Execution DLL File

We have included in the program several predefined Conditions and predefined Executions.

However if you need a special Condition or Execution that is not included in the program, you can add it yourself by including it in a special purpose DLL or "ActiveX DLL".

Sample code for the DLL can be found in the directory **SampleDLLs** under the QSetup directory.

The DLL must export at least 3 functions with the following prototype:

**C/C++ Code:**
```
__declspec(dllexport) int __cdecl GetDllVersion()
__declspec(dllexport) char* __cdecl GetConditionAlias(int n, char* Buf)
__declspec(dllexport) char* __cdecl GetExecutionAlias(int n, char* Buf)
```

**Pascal Code:**
```
function GetDllVersion: integer; cdecl;
function GetConditionAlias(n: integer; Buf: PChar): PChar; cdecl;
function GetExecutionAlias(n: integer; Buf: PChar): PChar; cdecl;
```

**Visual Basic Code:**
```
Public Function GetDllVersion() As Integer
Public Function GetConditionAlias(n As Long, Alias As String) As String
Public Function GetExecutionAlias(n As Long, Alias As String) As String
```

**C# Code:**
```
Public  int GetDllVersion()
public string GetConditionAlias(int n, string Dummy)
public string GetExecutionAlias(int n, string Dummy)
```

**VB.NET Code:**
```
Public Function GetDllVersion() As Integer
Public Function GetConditionAlias(ByVal n As Integer, ByVal _alias
As String) As String
Public Function GetExecutionAlias(ByVal n As Integer, ByVal _alias
As String) As String
```

The return value of the first "GetDllVersion" function must be **1**.

The DLL or "ActiveX DLL" may include up to 10 Conditions and up to 10 Executions - marked (0..9), with the following prototype:

**C/C++ Code:**
```
__declspec(dllexport) int __cdecl GetCondition_0(HWND Wnd, int
Stage, char* Arg1, char* Arg2, char* Arg3)
__declspec(dllexport) int __cdecl GetCondition_1(HWND Wnd, int
Stage, char* Arg1, char* Arg2, char* Arg3)
    ...
__declspec(dllexport) int __cdecl GetCondition_9(HWND Wnd, int
Stage, char* Arg1, char* Arg2, char* Arg3)


__declspec(dllexport) int __cdecl SetExecution_0(HWND Wnd, int
Stage, char* Arg1, char* Arg2, char* Arg3)
__declspec(dllexport) int __cdecl SetExecution_1(HWND Wnd, int
Stage, char* Arg1, char* Arg2, char* Arg3)
    ...
__declspec(dllexport) int __cdecl SetExecution_9(HWND Wnd, int
Stage, char* Arg1, char* Arg2, char* Arg3)
```

**Pascal Code:**
```
function GetCondition_0(Wnd: HWND; Stage: integer; Arg1,Arg2,Arg3:
PChar): integer; cdecl;
function GetCondition_1(Wnd: HWND; Stage: integer; Arg1,Arg2,Arg3:
PChar): integer; cdecl;
    ...
function GetCondition_9(Wnd: HWND; Stage: integer; Arg1,Arg2,Arg3:
PChar): integer; cdecl;


function SetExecution_0(Wnd: HWND; Stage: integer; Arg1,Arg2,Arg3:
PChar): integer; cdecl;
function SetExecution_1(Wnd: HWND; Stage: integer; Arg1,Arg2,Arg3:
PChar): integer; cdecl;
    ...
function SetExecution_9(Wnd: HWND; Stage: integer; Arg1,Arg2,Arg3:
PChar): integer; cdecl;
```

**Visual Basic Code:**
```
Public Function GetCondition_0(Wnd As Long, Stage As Long, Arg1 As
String, Arg2 As String, Arg3 As String) As Integer
Public Function GetCondition_1(Wnd As Long, Stage As Long, Arg1 As
String, Arg2 As String, Arg3 As String) As Integer
    ...
Public Function GetCondition_9(Wnd As Long, Stage As Long, Arg1 As
String, Arg2 As String, Arg3 As String) As Integer


Public Function SetCondition_0(Wnd As Long, Stage As Long, Arg1 As
```

```
String, Arg2 As String, Arg3 As String) As Integer
Public Function SetCondition_1(Wnd As Long, Stage As Long, Arg1 As
String, Arg2 As String, Arg3 As String) As Integer
   ...
Public Function SetCondition_9(Wnd As Long, Stage As Long, Arg1 As
String, Arg2 As String, Arg3 As String) As Integer
```

**IMPORTANT Note for VB Programmers**
The VB_Name attribute MUST BE: "Exec".
For instructions on how to compile the VB code please read the file
"HowToCompileSamples.txt" found in the VB directory.

## C# Code:

```
public int GetCondition_0(int Wnd, int Stage, string Arg1, string
Arg2, string Arg3)
public int GetCondition_1(int Wnd, int Stage, string Arg1, string
Arg2, string Arg3)
   ...
public int GetCondition_9(int Wnd, int Stage, string Arg1, string
Arg2, string Arg3)


public int GetCondition_0(int Wnd, int Stage, string Arg1, string
Arg2, string Arg3)
public int GetCondition_1(int Wnd, int Stage, string Arg1, string
Arg2, string Arg3)
   ...
public int GetCondition_9(int Wnd, int Stage, string Arg1, string
Arg2, string Arg3)
```

## VB.NET Code:

```
public int SetExecution_0(int Wnd, int Stage, string Arg1, string
Arg2, string Arg3)
public int SetExecution_1(int Wnd, int Stage, string Arg1, string
Arg2, string Arg3)
   ...
public int SetExecution_9(int Wnd, int Stage, string Arg1, string
Arg2, string Arg3)
```

# Check Dependency

When you deliver your program to a customer you must make sure you also deliver all the DLL and OCX files that your program depends on.
The "Check Dependency" option will help you find out what DLLs and OCXs are required.

DLLs may be linked to your program in 2 ways Statically & Dynamically.

**Static Linking**
Static Linked DLLs are those linked to your program by the Compiler or Linker at Compile time.
Those files may be identified by performing a static check. In this check the EXE file is scanned and the names of the linked files are read from the file's header.
No need to actually run the executable.

**Dynamic Linking**
Dynamic Linked DLLs and OCXs are those linked to your program at run time usually using the "LoadLibrary" API call. To identify those files, the scanner must RUN your program and identify each DLL or OCX as your program attempts to load it at runtime.

To learn more about the theory of the subject goto: www.dependencywalker.com.

## Static Check
**Static Check All**
When you click this button the scanner will perform a "Static Check" on all the EXE, DLL & OCX files found in your setup.
You can browse the list of all avaiable files by clicking the arrow on the "File Name" ComboBox.
All Statically linked files will be displayed in blue.

**Static Check**
When you click this button the scanner will perform a "Static Check" only on the file that its name is displayed in the "File Name" ComboBox.
All Statically linked files will be displayed in blue.

## Dynamic Check
**Start**
When you click this button the scanner will perform a "Dynamic Check" on the file that its name is diplayed in the "File Name" ComboBox.
The scanner will run the file and monitor any calls comming from inside the executable while it is running.
At this point of time you should start PLAYING with the program in such a way that it will go through all its options to make sure that any Dynamic linked file is actually loaded.
If needed you may add "Parameters" to run the executable. (Parameters are data items like document name that you normally add on the command line when you start an application).
All Dynamically linked files will be displayed in red.

**Stop**
Click this button to Stop the "Dynamic Check" test.

**Important Files**

The scanner will list all the linked files when it runs. However many of those files are Windows generic and need not be included in your setup. You should include in your setup only those files that your program is adding. The scanner will do its best to identify the Important files for you. All important files will be displaed with BOLD font and their CheckBox will be checked.
You may check or uncheck any file later as needed.

**Add Files**

After the scan is finnished click the **[Add Files]** button to add the new files to your installation delivery. Only files that are checked will be added. The files will be added to the current directory, you can then move them to another directory using Drag & Drop.

# Advanced Auto Update

The following section explains how you can achieve better control of the Auto Update process - using direct communication between your **Application** and the **Agent**.
To take advantage of the options listed here you must be an experienced Windows programmer with solid understanding of "Windows Messages" programming.

## The Concept

The "Auto Update" process as described so far is designed to work automatically with minimum intervention of the Application that is being updated. In the following section we will explain how you can achieve better control of the "Auto Update" process from within your running Application. Basically control is achieved by exchanging messages between the running Application and the Agent.

## Establishing a Communication Channel

Communication is performed using Windows Messages.
The Application will send a Windows Message to the Agent using **SendMessage()** or **PostMessage()**.
The Agent will send a Windows Message to the Application using **SendMessage()** or **PostMessage()**.

The process is initiated by the Application.
The Application defines a Message (Let's call it MsgApp).
The Application defines a Window handle (Let's call it WndApp).
The Application informs the Agent of this two data items using **WinExec()**.

From now on when ever the **Agent** needs to send information to the **Application** it will use a command like this:
    SendMessage(WndApp, MsgApp, wParam, lParam);
    Actual data will be carried in wParam & lParam.

In Response to the **WinExec()** command the Agent will send 2 messages back to the Application.
The first message will carry the Window's handle of the Agent (Let's call it WndAgnt).
The second message will carry the defined Message of the Agent (Let's call it MsgAgnt).

From now on when ever the **Application** needs to send information to the **Agent** it will use a command like this:
    SendMessage(WndAgnt, MsgAgnt, wParam, lParam);
    Actual data will be carried in wParam & lParam.

**To establish a communication channel perform the following operations:**

- Define a special Windows Message that is unique to your application. We suggest that you define a message in the range 1200..1300 (Decimal).

- Add to your main Windows Procedure an entry that will be sensitive to this message.

- Call the Agent with a command line similar to the following:
  WinExec("PhoneBookUpdate.EXE /WM_MSG=WWW:MMM",0);
  WWW = WndApp (Decimal value).
  MMM = MsgApp (Decimal value).

- The agent will respond with 2 messages sent to the WndApp.
  - The First message will have the following data:
    wParam = AU_InformAgentHWND (50).
    lParam = The HWnd of the Agent.
    You must record the lParam in your application.

  - The Second message will have the following data:
    wParam = AU_InformAgentMSG (51).
    lParam = The special message you will use when you call the Agent.
    You must record the lParam in your application.

At the end of this process, the Application knows how to send messages to the Agent, and The Agent knows how to send messages to the Application.

## Message Structure

Following Windows conventions, every Message include 4 items of information: **HWnd, Message, wParam** & **lParam**. A message may also return a **Result** code.
We use those items as follows:
- **HWnd** - the HWindow to which the Message is being sent (WndApp or WndAgnt in our case).
- **Message** - The Message identifier (MsgApp or MsgAgnt in our case).
- **wParam** - Instruction/Request.
- **lParam** - Additional data if required.
- **Result** - Optional return code if required.

## Instructions/Requests From Application to Agent

The following list describes all the Requests and Instructions the **Application** can send to the **Agent.**

**Example:**
To ask the Agent what is the Version number as stored in the .ORIGINAL file, issue the following message:

**Res = SendMessage(WndAgnt,MsgAgnt,4,0);**
Upon return the **Res** variable will hold the required information.
In this example **4** is the **AU_GetVersionOriginal** instruction contained in wParam parameter.

**AU_GetVersionOriginal**
wParam:    4
lParam:    0
Result:    VersionNum
Description:    Return the version number as stored in the .ORIGINAL file.

**AU_GetVersionInfo**

wParam:     5
lParam:     0
Result:      VersionNum
Description:    Return the version number as stored in the .INFO file.

### AU_GetAgentStatus
wParam:     7
lParam:     0
Result:      0=Idle, 1=DLInfo, 2=DLData, 3=Updating
Description:    Return the current status of the Agent.

### AU_GetProbeSecondsInterval
wParam:     8
lParam:     0
Result:      Value in Seconds
Description:    Return the Probe interval in seconds from the .ORIGINAL file.

### AU_GetReAskSecondsInterval
wParam:     9
lParam:     0
Result:      Value in Seconds
Description:    Return the ReAsk interval in seconds from the .ORIGINAL file.

### AU_GetProbeSecondsLast
wParam:     10
lParam:     0
Result:      Value in Seconds
Description:    the last time the Agent probed for new version (counted in
        seconds from 12/30/1899 12:00 am).

### AU_GetReAskSecondsLast
wParam:     11
lParam:     0
Result:      Value in Seconds
Description:    Return the last time the Agent Asked for permission to update
            (counted in seconds from 12/30/1899 12:00 am).

### AU_SetProbeSecondsInterval
wParam:     12
lParam:     0
Result:      0
Description:    Set the Probe interval in seconds in the .ORIGINAL file.

### AU_SetReAskSecondsInterval
wParam:     13
lParam:     0
Result:      0
Description:    Set the ReAsk interval in seconds in the .ORIGINAL file.

**AU_SetProbeSecondsLast**
> wParam:    14
> lParam:    0
> Result:    0
> Description:    Set the last time the Agent probed for new version (counted in
>            seconds from 12/30/1899 12:00 am).

**AU_SetReAskSecondsLast**
> wParam:    15
> lParam:    0
> Result:    0
> Description:    Set the last time the Agent Asked for permission to update
>        (counted in seconds from 12/30/1899 12:00 am).

**AU_SetRequestConfirmationBeforeTest**
> wParam:    20
> lParam:    0=false 1=true
> Result:    0
> Description:    Instruct the Agent to Request for confirmation before testing
>        for new update YES/NO.

**AU_SetRequestConfirmationBeforeDownload**
> wParam:    21
> lParam:    0=false 1=true
> Result:    0
> Description:    Instruct the Agent to Request for confirmation before Download
>            YES/NO.

**AU_SetRequestConfirmationBeforeInstall**
> wParam:    22
> lParam:    0=false 1=true
> Result:    0
> Description:    Instruct the Agent to Request for confirmation before Install
>        YES/NO.

**AU_SetPerformBackGroundUpdate**
> wParam:    23
> lParam:    0=false 1=true
> Result:    0
> Description:    Instruct the Agent to perform the update in the background
>        YES/NO.

**AU_SetInformUserWhenUpdateFinished**
> wParam:    24
> lParam:    0=false 1=true
> Result:    0
> Description:    Instruct the Agent to inform the user when update is finished
>        YES/NO.

**AU_SetRestartProcessWhenUpdateFinished**

wParam:      25
lParam:      0=false 1=true
Result:      0
Description:    Instruct the Agent to restart the application when Update is
   finished YES/NO.

### AU_PerformStopAgent

wParam:      30
lParam:      WaitTime in Seconds
Result:      0
Description:    Instruct the Agent to Stop itself lParam seconds from now.

### AU_PerformUpdateCycle

wParam:      31
lParam:      0
Result:      0
Description:    Instruct the Agent to perform a complete update cycle 1. Probe
   for a new .INFO file, if a file is found, analyze it and download
   the DATA file, and continue to perform Update.

### AU_PerformDownloadInfo

wParam: 32
lParam:   0
Result:    0
Description:      Instruct the Agent to download the .INFO file.


## Instructions/Requests from Agent to Application

The following list describes all the Requests and Instructions the **Agent** can send to the
**Application**.

### AU_InformDownloadInfoFile

wParam:      40
lParam:      1=Start 2=End
Result:      0
Description:    Inform the Application on Start and End of .INFO file download.

### AU_InformDownloadDataFile

wParam:      41
lParam:      1=Start 2=End
Result:      0
Description:    Inform the Application on Start and End of DATA file download.

### AU_InformUpdateProcess

wParam:      42
lParam:      1=Start 2=End
Result:      0
Description:    Inform the Application on Start and End of the UPDATE
   process.

**AU_InformAgentIsGoingDown**
  wParam: 43
  lParam:  0
  Result:  0
  Description: Inform the Application when the Agent is closing itself.

**AU_InformAgentHWND**
  wParam: 50
  lParam:  0
  Result:  0
  Description: Inform the Application on the HWindow to send messages to (MsgWnd).

**AU_InformAgentMSG**
  wParam: 51
  lParam:  0
  Result:  0
  Description: Inform the Application on the MsgAgnt .

# FTP Upload

Use this option to UPLOAD your setup file(s) to the Internet directly from the Composer, using FTP protocol.

## Connection Data:

To establish connection with your website using FTP protocol you must supply the following login data items:
- Hostname
- Username
- Password

After you enter the data click the **[Directory]** button.
If the connection is successful, you will see the root directory of your website and all its child directories. Click the directory names to navigate to the directory where you want to store your Setup file(s) in.
You can use the **[MK Dir]** & **[RM Dir]** buttons to create & remove directories.

**FTP Port**
According to the FTP Standard, the default FTP port is 21 (Decimal).
If your FTP Server is using another port, you can specify the port by adding a colon and the new port number to the **Hostname**.
**Example -** ftp.microsoft.com:1234

## Files to Upload:

The Upload dialog can be called from 2 places:
- The main **[Upload]** button - at the bottom-left of the Composer screen.
- The **[Upload]** button on the "Auto Update" page.

In every option, a different list of files is presented for uploading. You can control which file will be actually uploaded by Checking/UnChecking the relevant CheckBox.

## Upload

Click the **[Upload]** button to upload your setup file(s) to the Internet.

**PLEASE NOTE**

- In most FTP client programs, you must establish connection before you can upload your files. In QSetup the **[Upload]** button will perform the "*Connect - Upload - Disconnect*" cycle automatically with one click.

- For security reasons, all the connection data you enter in this dialog (Hostname, Username, Password, etc...) is stored in the registry of your PC (not in the QSP file).

# Adding new language to QSetup

To add new language to QSetup create a new language file. The file must have a name that reflects the language (French.lng, German.lng, Spanish.lng, etc...).

The recommended way to do it, is by copying the "English.lng" file to the new file and edit the new file as required.

The language file is a text file in the form of an INI file and should be edited with an ASCII plain text editor like NOTEPAD.

After translation, place the file in the LANG directory, and restart QSetup.

The first section of the language file is called "General" and include the following data:

```
[General]
Version=2.0
Language=English
LangID=9
Charset=0
Author=Pantaray
Date=11-DEC-2004
```

If for instance you want to create a file for the Russian language, the file should be called "Russian.lng", and the first section should look like this:

```
[General]
Version=2.0
Language=Russian
LangID=25
Charset=204
Author=<Your Name>
Date=<Date>
```

**Following is a list of LangIDs as defined by Windows**

```
AFRIKAANS    = 54
ALBANIAN     = 28
ARABIC       = 1
BASQUE       = 45
BELARUSIAN   = 35
BULGARIAN    = 2
CATALAN      = 3
CHINESE      = 4
CROATIAN     = 26
CZECH        = 5
DANISH       = 6
DUTCH        = 19
ENGLISH      = 9
ESTONIAN     = 37
FAEROESE     = 56
FARSI        = 41
FINNISH      = 11
FRENCH       = 12
```

```
GERMAN        = 7
GREEK         = 8
HEBREW        = 13
HUNGARIAN     = 14
ICELANDIC     = 15
INDONESIAN    = 33
ITALIAN       = 16
JAPANESE      = 17
KOREAN        = 18
LATVIAN       = 38
LITHUANIAN    = 39
NORWEGIAN     = 20
POLISH        = 21
PORTUGUESE    = 22
ROMANIAN      = 24
RUSSIAN       = 25
SERBIAN       = 26
SLOVAK        = 27
SLOVENIAN     = 36
SPANISH       = 10
SWEDISH       = 29
THAI          = 30
TURKISH       = 31
UKRAINIAN     = 34
VIETNAMESE    = 42
```

**Following is a list of Charsets as defined by Windows**

```
ANSI_CHARSET        = 0
DEFAULT_CHARSET     = 1
SYMBOL_CHARSET      = 2
SHIFTJIS_CHARSET    = 128
HANGEUL_CHARSET     = 129
GB2312_CHARSET      = 134
CHINESEBIG5_CHARSET = 136
OEM_CHARSET         = 255
JOHAB_CHARSET       = 130
HEBREW_CHARSET      = 177
ARABIC_CHARSET      = 178
GREEK_CHARSET       = 161
TURKISH_CHARSET     = 162
VIETNAMESE_CHARSET  = 163
THAI_CHARSET        = 222
EASTEUROPE_CHARSET  = 238
RUSSIAN_CHARSET     = 204
```

**Please note**
All West European langauges must use the ANSI_CHARSET (0).
All East European langauges must use the EASTEUROPE_CHARSET (238).

**The following tags have special meaning**

<P> means insert a line-break.
<T> means insert a Tab.
<B> means Start BOLD text.
</B> means End BOLD text.
PROG_NAME will be replaced by the "Program Descriptive Name".

**ATTENTION**
When translating make sure you comply with the INI file
standard - that is:
1. Do not change the names of the sections.
2. Do not change the names of the items.
3. Do not insert RETURNS in the middle of a line.

**REQUEST**
Please send us a new language file you created, so that we can
add it to the program for the benefit of all our users.

Send the file to: support@pantaray.com

# How to compile the samples using Visual Basic?

1. Create a new "ActiveX dll" project by clicking new and selecting "ActiveX dll" from the "new project" dialog.

2. From the visual basic "Project" menu select "Add file" and choose the .cls file supplied.

3. Open the project properties "Project|Project properties" from the Menu and set the project name to "VBSerialCheck" or "VBExec".

4. Delete the extra unnecessary class1 from the project.

5. Make the ActiveX dll by choosing File|Make ... project menu item.

# Custom Dialogs

QSetup is provided with a list of 12 predefined dialogs. The dialogs are listed on the "Dialogs" page and you can select which Dialogs will be displayed to your customers during the setup process.

QSetup also offers you the option to define one or more Custom Dialogs that will suite the special needs of your setup.

To define custom dialogs click the **[Custom Dialogs...]** button on the "Dialogs" page.

## Custom Dialogs Designer

When you click the **[Custom Dialogs...]** button the special "Custom Dialogs Designer" tool will open.

Using this tool you can create as many dialogs as you wish. Each Dialog will contain as many controls as needed from an assortment of the 11 most popular controls available under Windows.

Once you define a dialog this dialog will be added to the list of dialogs found on the "Dialogs" page.

Using this list you can do the following:
- Set the order of the dialogs.
- Set an image to the top/right corner of the dialog.
- Disable the dialog.

The "Custom Dialogs Designer"  tool have 2 main areas:
- Controls
- Dialogs

## Dialogs Area

This area looks very similar to the regular dialog when using the "Modern" dialog style.

**New**

To create a new dialog click the **[New]** button. In the screen that opens enter the "Dialog Name". Please note that the name you enter here will later be used as the title of the dialog. The name you entered will be added to the list of dialogs in the ComboBox and also displayed as the dialog title.

**Rename**

Click this button if you want to modify the name of the current dialog.

**Delete**

Click this button if you want to delete the current dialog.

**Description**

Click this button to enter a more detailed description of the purpose of the dialog. This description will be displayed in bold on the top white area of the dialog.

**Adding Controls**

To create a dialog you merely add controls to it. Adding controls is done from the **Controls Area**. When you add a control it will be placed on the Top/Left corner of the dialog.

Once a control is added you can do the following:
- Move the control to any place on the form using the mouse or with the keyboard.
- Resize the control using the mouse or with the keyboard.
- Set various properties of the control, using the **Controls Area**.

**Dialog Names ComboBox**
You may add several dialogs to a setup. When adding a new dialog its name will be added to the "Dialog Names ComboBox". Using this ComboBox you can scroll through the different dialogs.

**Modern/Classic Dialog Style**
QSetup offers 2 basic setup dialog styles: Classic & Modern.
The "Custom Dialog Designer" is designed after the Modern dialog style.
We recommend that when you add custom dialogs to your setup you will use the "Modern Dialog Style".

You can still add Custom Dialogs to a "Classic Dialog Style", however you will have to take extra care when adding controls and place them at the Top/Left area of the Dialog form.

# Controls Area

Using this area you will add controls to your Custom Dialog.

**Add**
To add a control click the **[Add]** button. The "Add Control" screen will open.
In this screen you can select any of the following controls:
- Label
- GroupBox
- Button
- Memo
- Edit
- Panel
- CheckBox
- RadioButton
- ComboBox
- RadioGroup
- ListBox

Once you select a control, it will be placed on the current Dialog form, and its properties will be displayed in the **Properties** area.

**Properties**
The following properties are common to all controls:

- **Name** - The name of the control, MUST be unique.
- **OnClick/OnChange** - Here you will enter the action required when the control is clicked or changed (if needed).
- **X** - The X location of the control on the dialog form.
- **Y** - The Y location of the control on the dialog form.
- **W** - The width of the control.
- **H** - The height of the control.
- **Text** - The text or title (caption) of the control.
- **Bold** - Set this property to "True" if you want the text to be displayed in Bold.
- **Visible** - Set this property to "False" to hide the control.
- **Enabled** - Set this property to "False" to disable the control.
- **BidiEnabled** - When this property is set to "True" and the language of the setup is "Hebrew" or "Arabic" the control will change to RightToLeft orientation.

The following properties are available in some of the controls only:

- **Password** - (Edit control)
  Set this property to "True" to display entered characters as *.

- **Bevel** - (Panel control)
  Set this property to "Raised" or "Lowered", this will define the border of the Panel.

- **Checked** - (CheckBox control)
  Set this property to "True" to make the CheckBox "Checked".

- **ExclusiveGroup** - (CheckBox, RadioButton)
  The default value of this property is 0 (zero).
  When this property is set to Zero, the relevant CheckBox/RadioButton has no influence on the other CheckBoxes/RadioButton in the dialog.
  If you have some CheckBoxes/RadioButtons on your dialog and all have the same "ExclusiveGroup" value which is greater the 0 (zero) then when you click one CheckBox/RadioButtons all the others will be UnChecked.
  CheckBoxes have no influence on RadioButtons and vice versa.

- **Strings** - (ComboBox, RadioGroup, ListBox)
  Enter here some text values separated by a comma.
  Sample: COM1,COM2,COM3,COM4.
  All the items you entered will be available in the control for selection.

- **ItemIndex** - (ComboBox, RadioGroup, ListBox)
  This is an integer value that starts from 0 (Zero). This value indicates the number of the the selected item.
  If the values for selection are: COM1,COM2,COM3,COM4 and "COM3" is selected then "ItemIndex" will have the value of 2.

- **WordWrap** - (Label control)
  When this property is set to "True" the text of the Label will wrap if it is too long.
  When this property is set to "True" the system will take into consideration the W & H properties when calculating the area of the lable text.

- **MinimizePath** - (Label control)
  When this property is set to "True" the text of the Label will minimize if it is too long.
  This property is valid only if the text in the lable represents a file Path and the path includes the backslash character.
  When this property is set to "True" the system will take into consideration the W propertie when calculating the area of the lable text.

All properties except **Name** can be Edited/Modified in the Properties list.
All properties can be read at setup time using the "Execute Engine".
All properties except **Name & OnClick** can be written at setup time using the "Execute Engine".

**Multiline Text**
The Memo & Label controls can accept Multiline text.
To add a Line-Break enter the **<P>** alias.
To add a Tab enter the **<T>** alias.

**Moving a Control using the Mouse**
Move the mouse to the center of the control.
When the mouse pointer change to a finger icon, press the left mouse button.
Drag the control to the required location.
Release the mouse.

**Sizing a Control using the Mouse**
Move the mouse near the edge or corner of the control.
When the mouse pointer change to the proper arrow, press the left mouse button.
Move the mouse until the control reach the required size.
Release the mouse.


**Moving a Control using the Keyboard**
Click the required control to select it.
Click the Arrow keys while holding the "Ctrl" key.

**Sizing a Control using the Keyboard**
Click the required control to select it.
Click the Arrow keys while holding the "Shift" key.

You can also move or size a control by directly editing its X,Y,W,H properties.

**Rename**
To rename a control click the **[Rename]** button.
When you enter a new name observe the following rules:
1. A name must be unique in all the dialogs.
2. The name can have only Alpha Numeric characters and the '_' character.
3. Names are NOT case sensitive.

**Order of Controls**
Using the 2 Arrow Buttons, you can move a control up or down in the controls list.
The order of a control will define its visibility when controls are placed one on top of the other.
A "Later" control will be placed on top of a "Previous" control.


## Preview! Button

Click the **[Preview!]** button to enter "Preview" mode.
In this mode you will see how the dialog will look like during actual installation.
If you have already created several dialogs click the **[Next >]** and **[< Back]** buttons to scroll through the dialogs.
To exit Preview mode click the **[Preview!]** button once again or click the **[Cancel]** button.


## Bidi Test CheckBox

When this option is checked, all controls that have their BidiEnabled property set to True will change to RightToLeft orientation.


## Save Button

Use this button to save the dialog information. When clicking this button the information will be added to the QSP file, and the QSP file immediately stored to the disk.
Also the names of the newly added Dialogs will appear in the dialogs list of the "Dialogs" page highlighted with blue color.

## Tools Button

Click this button to select any of the following service operations:

**Copy Control**
Copy the current control to the clipboard.

**Paste Control**
Paste a control from the clipboard to the current dialog.

**Copy Dialog**
Copy the current dialog to the clipboard.

**Paste Dialog**
Paste a dialog from the clipboard to the tool, thus adding a new dialog.

**Export Dialog**
Save the current dialog to a file, the file name will have the extension *.qspdlg.

**Import Dialog**
Read a dialog file from the disk into the tool, thus adding a new dialog.

**Align to Grid**
When this option is checked (default) all controls will be aligned to a virtual grid when moved with the mouse. This option has no effect when moving the control using the keyboard or by direct editing of the X,Y,W,H properties.

**Grid**
Select the virtual Grid resolution. You can select any of the following values: 2,3,4,5,6,8,10,12,14,16. We recommend that you use the value of 4 (default). This value provides the best compromise between detailness and ease of use.

**ToolTip**
When this option is checked and you move the mouse over a control, the control coordinates will be displayed in the mouse tooltip.

**Language Suppor**t
Select **Language Support**... to open a special "Language Support" dialog. Using this dialog you can translate all the phrases of the custom dialogs you created, to every language you want to support in your setup.
To test the influence of the language support on your Cusom Dialogs you must close the "Custom Dialogs Designer" and click the [Preview] button on the Bottom/Right of the Composer.

**IMPORTANT**
To properely test your multilingual setup you must adjust your operating system to the language under test.
For more info read the following link: www.pantaray.com/language.html#testing.


## Special Controls

**Edit**
The height of an Edit control is defined by the operating system, thus the H property of the Edit control will be set initially to Zero.
You can manually change this value but it will have no effect on the control behavior.

**Label**
The height & width of a Label control are automatically defined at runtime, thus the H & W property of the Label control will be set initially to Zero.
You can manually change the H value but it will have no effect on the control behavior.
If you set the W value of a Label to a value greater the Zero and you set the property **BidiEnabled** to True then when "Bidi Test" is Checked the text in the label will flip to the right according to the W value of the control.


## Interacting with the Custom Dialog

Interacting with a Custom Dialog at setup time is done using the "Execute Engine".


## OnClick

Every Control can be linked to an "Execute Item" using the OnClick event.

Lets say you have a button on your Custom dialog.
When the user click this button at setup time you want some action to take place.

To program this operation do the following:

1. Go to the "Execute" page and click **[Add Item...]**.
2. Select "Undefined Stage" in the "Perform At" selection box.
3. Program the execute item according to your needs.
4. Click **[Apply]** and close the "New Execution Item" dialog.
5. Go to the "Dialogs" page and click **[Custom Dialogs...]**.
6. Add the required button.
7. Click the selection box of the "OnClick" property.
8. Select the "Execute Item" you just created.
9. Click the **[Save]** button.

To test this operation do the following:

1. Close the "Custom Dialog" screen.
2. Compile your setup by clicking the **[Compile]** button.
3. Run your setup by clicking the **[Run]** button.
4. Click the **[Next >]** button until you reach the Custom dialog.
5. Click the required button on the custom dialog.

## Before/After Dialog

Using this option you can define operations that will take place just before and/or just after a Custom Dialog is displayed.

Lets say you created a Custom Dialog by the name "Dialog Number 1".

Go to the "Execute" page and click **[Add Item...]**.
Give the new Item a name and click the "Perform At" selection box.
At the end of the list you will see the following 2 lines on yellow background:
1. "Before Dialog Number 1"
2. "After Dialog Number 1".

If you select the first option, the Execute Item will be preformed just before the Custom Dialog is displayed.
If you select the second option, the Execute Item will be preformed just after the Custom Dialog is displayed.

## Read/Write Control Properties

Reading and Writing control properties at setup time can be done with special commands from the "Execute Engine".

The following "Conditions" are available:
- Control Text is
- Control Item Index is
- Control is Checked

The following "Executions" are available:
- Set Control Property
- Get Control Property

### Condition Samples:

**Control Text is**
> Argument-1: Edit2
> Comparison: =
> Argument-2: Bill Gates

**Control Item Index is**
> Argument-1: ComboBox3
> Comparison: >=
> Argument-2: 2

**Control is Checked**
> Argument-1: CheckBox1

### Execution Samples:

**Set Control Property**
> Argument-1: Edit2
> Argument-2: Text
> Argument-3: Bill Gates

**Get Control Property**

Argument-1: Edit2
Argument-2: Text
Argument-3: <UserName>