# Welcome to the help pages of WinBuilder

Inside this manual you'll find details and quick start guides related to WinBuilder. If you haven't found enough information to answer your questions, please let us know what you think needs to be changed or added to make this manual easier to understand.

You are kindly invited to place your comments here or vist us at the main WinBuilder forum.

# **Overview & Getting Started**

WinBuilder is a flexible scripting environment that is well suited to building boot disks and other Windows "Pre-installation Environment" (or PE) images. Using special scripts, this program can create different bootable environments based on the source that is used.

There are several <u>projects</u> available for WinBuilder so please remember to read the instructions found inside each project to understand which sources are compatible and can be used with that project.

Some of the features to expect from WinBuilder include:

- Many built-in tools to edit and create new scripts
- Friendly User Interface
- Scripts that in most cases, can be configured without needing to edit any code (i.e Scriptable GUI components)
- Most scripts already come included with freeware applications
- Scripting Language and API specially targeted at Windows PE challenges.
- Last but not least A very active community

Getting started with WinBuilder is easy, since there are many projects that have done most of the work for you. Jump over to the <u>projects section</u> of this documentation or <u>directly</u> to the !WinBuilder Project fforums for more details.

# The WinBuilder Scripting Language

The behavior of WinBuilder is driven by it's own internal scripting language. These "Script files" are simple text files that use the extension .script. They are most easily edited with WinBuilder's internal editor, but you can also use your favorite text editor or a simple text notepad.

Script files use a very simple to understand language to run their commands and actions. In general, the syntax is very simple - i.e. keywords for functions with the parameters separated by commas. Comment lines start with two slashes, and are skipped during processing. Out of compatibility concerns two hash marks are accepted as comment marker.

#### Example:

```
// This is a comment line
## This is a comment line, too
```

Settings are written in the form of Key=Value (e.g. Color=Blue) and these values are kept inside sections similar to the structure of an \*.ini file (e.g. [MySection]).

You can also add additional files inside your scripts. These files are stored as attachments in a process very similar to the way email messages used to include binary files - i.e. they are "encoded" into an ASCII format. This way you can share your scripts in discussion forums with all the needed tools inside.

Script files can perform both specific and generic functions. Some scripts will be responsible for creating the base structure of the project while others will be used to add extra functionality and programs.

These scripts can also use a very specific model that only works on a single project and cannot be successfully used on another project unless it also shares the same model as the original project from where it came. The model consists of functions and variables stored in scripts.

# Syntax Rules and Escape Characters

#### First some rules:

- 1. WinBuilder Script Line Parameters are separated by a comma. If a comma is inside an argument, it must be 'escaped' by #\$c
- 2. Besides some special cases (e.g. parameters in <u>ShellExecute</u>) generally enclosing quotes are not needed.
- 3. Enclosing quotes MUST be used if the WinBuilder Script Line Parameters contain spaces.

Since there are times when the comma or quote need to be used inside a string or parameter, WinBuilder has a character escape mechanisim. The following are the available escape characters:

- #\$c Comma (, )
- #\$p Percent (%)
- #\$q Double quotes (")
- #\$s Space ( )
- #\$x Newline (<cr/1f>)
- #\$z String separator (i.e. the unicode 0 which is only used by the RegWriteBin command)

Here is a short explanation of what can be done and what cannot be done and when the escaped characters should be used.

- If the spaces are replaced by the escape #\$s, the enclosing quotes should be omitted.
- In most cases user violation of the rules are accepted by WinBuilder and processed as assumed. E.g. unnecessary enclosing quotes or a comma inside an item enclosed by quotes.

But sometimes that can bring unexpected results. Let me give an Example:

Item1,item2 is understood as two items, separated by the comma.
Item1#\$citem2 and "Item1,item2" are both correctly written as ONE item with the comma inside.

Now the problem occurs when it is not possible to determine what is meant. For example, **Set**,%**Var**%, "**Item1**", "**item2**" cannot be understood uniquely by WinBuilder?. Let's exchange the English understandable words by the character 'x'

#### "xxxxx", "xxxxx"

- Are there two quote-enclosed xxxxx, separated by a comma?
- Or is there on long string of 10 xes, interrupted by a nested quote pair containing a comma?
- Or is there on long string of 10 xes, interrupted by a nested quote pair 'escaping' a comma? > Set,%var%,xxxxx#\$cxxxxx

You see how important it is to follow the rules, even if WinBuilder understands most violations correctly.

### File Structure

Let's start by looking at the structure of a typical script file. We use sections to group commands and settings inside script files.

- The [Main] section
- The [Variables] section
- The [Process] section
- The [Interface] section
- The [OnProcessEntry] section
- The [OnProcessExit] section

The easiest way to see how they are applied is to edit a few scripts and look at the source code to learn more about them. A typical script is composed of at least these 3 important sections

- Main will keep all details that identify your script
- Variables used to keep variable values
- Process where your commands are placed

In addition to the script files, there is a master project file called Script.project, which has two optional sections

- OnProcessEntry run as first command of every project
- OnProcessExit run as last command of every project

Here is a small example of how it should look:

```
[Main]
Title=Add my Program
Description=This script will add my program
Level=5

[Variables]
%myTitle%="This is a title for the program"

[Process]
echo,%myTitle%
....
....
```

We can additionally use more sections to add variables from the script interface or even specify more sections with commands to be processed.

#### The [Main] section

Inside this main section you can store the definitions that identify your script and allow it to be properly categorized inside the project. This section works like a normal INI section where you write all your data in the form of Key=Value (e.g. Color=Blue)

#### List of available keys

- Title Title that is displayed by your script
- Description a short description of what the script is intended to do
- Author the author or group involved in the initial release, following releases or branches should be mentioned in Credits
- Credits details the people involved making the script possible or persons involved in updating the work.
- Date date of changes, any text is accepted recommended to use "Year-Month-Day" all written in numeric format
- Version scripts support version and they are important to mark new updated scripts on the download center
- Website / topic discussion place a link here to help users to find their way to a site with more informations
- Download\_level used by download center to categorize scripts under download profiles (1..4) See the documentation on the <u>Download Center</u> for more information.
- Selected using True/False will make your be checked or not for being processed. These are usually set (checked or not) using the !WinBUilder GUI. To completely remove any checkbox, set the value to "None". eg. Selected=None
- Locked use True/False to allows the users to edit the script interface
- Level very important setting to allow your script to be placed on the correct sequence when the project is running (see <a href="Script Levels">Script Levels</a> for more details on this)
- Interface Can be used in complex script to select a specific section of the script file to become the interface. If not specified, the displayed interface will be driven by the [Interface] section. See <u>this page</u> for more information

While some of the keys can be manipulated using the WinBuilder GUI, All of these keys can be edited easily using WinBuilder's internal source code editor. And while these keys are optional, it is recommended to place at least a few keys to better describe your script to others (i.e. Title and Description).

• **Note1:** Since WinBuilder will ignore any keywords not mentioned above for a specific purpose, some script authors also use the "INI" style format to add version specific history and other information. Just be sure that each key is unique. For example:

```
History01=This is a note about version one
History02=This is about version two
Note001=Another note
```

• **Note2:** If they are not found inside the script, the following values will be used (i.e. default values

```
Title=<Uses the filename of the script>
Level=5
Download_Level=2 (i.e. Recommended)
```

#### Added in 2009 (WB78?)

• **Certification** - To identify if a script has been "changed", the concept of script certification was added. When a project 'owner' uploads a script, he can give a 'stamp' to the script which is unique. Every change of the script can then be detected. It was enhanced to be smart enough to allow some GUI options changes, but not changes to the logic or functionality.

In the log it is easy to be seen when the script has been changed in core areas.

```
() > script does not have the stamp
(+) > script is still original
```

(?) > script has been changed.

Since this is a tool exclusive for 'project-owners' to certify their scripts, the keywords below are generated by a separate, console program to 'certify' the script before upload. In addition to the certify.exe program a personal key file <name>.wbc is also required. It writes something like this into the [Main] section:

```
[Main]
...
CertifiedBy=xxx
Certification=cd55abf6082298a0044f110479f98b7e
```

In addition to the information shown above, WinBuilder will show who issues the certification, or give a warning if there is no certification information in the script.

### The [Variables] section

This section is described in more detail on a separate page.

### The [Process] section

All commands you write in this section will be executed by the program. From this section you can also call other sections from the same script or other scripts if necessary using the <u>run command</u>.

The result of each operation is written on the log window - which you can later view or save to an html page.

### The [Interface] section

While this is important to scripts it is described in more detail **on a separate page**.

### The [OnProcessEntry] section

This section is an optional section exclusive in script.project. If it is present, during build this section is processed before the first [Process] line of every processed script.

### The [OnProcessExit] section

This section is an optional special section exclusive in script.project. If it is present, during build this section is processed after the last [Process] line of every processed script. For example, this will cause a specific section of the project.script file to be run when the build process exits.

[OnProcessExit]
Run,%ProjectDir%\script.project,Process-log

# **Script Levels**

Levels are used by WinBuilder to define when your script should run while the project is being built. This allows you to ensure that your script always runs at the correct time.

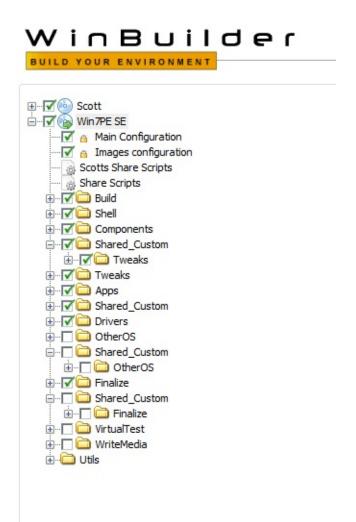
You can select values between 1 and 10, when the project starts it will begin with the scripts that have the lower level number and will increase a level after all scripts with the previous levels are processed. To further organize your scripts you can also create sub folders and in this way separate these scripts from others with the same level.

Winbuilder will process scripts in the following order:

- 1. Script Level
- 2. Folder Name
- 3. Script Name

File and folder names are processed in alpha numeric order. This information is used to build the "TREE" oriented view in the WinBuilder interface. While it might appear that this is just a simple folder/script view of the project - **IT IS NOT**.

Folders may be duplicated if there are script files of different levels inside the same folder. In the example below, notice how the "Shared\_Custom" folder is listed many times. This is because there are scripts in (or below) that folder level on disk that are at different levels.



Visually, this "Tree" view gives you a graphical representation of how WinBuilder will process your scripts. Top levels are usually used to be build the structure of your project, while the last levels are the ones where your project is finished and an image is created.

By default, we use level 5 as the middle value where most scripts that are only meant to add programs (i.e. <u>Application Scripts</u>) should be processed. Below is a proposed description of the build phases typically performed by each project:

- 1. Preprocess info (gathering information, mounting wims, etc.)
- 2. Build (basic building scripts create folders, copy/expand, winsxs)
- 3. Base (shell, shortcuts, ramdisk/FBWF options..)
- 4. Settings and Drivers (all tweaks and drivers)
- 5. Applications (added programs like CD-burn, editor, tools..)

- 6. PostProcessing (wim creation, autoUPX, cleanup)
- 7. ISO creation (mkisofs, RAM boot..)
- 8. PC Emulation (Qemu, vmware, virtualPC)
- 9. Burn ISO
- 10. Project Tools (hive editing, target tweaking..)

Corresponding negative script levels also exist with the only difference being that scripts with negative levels are not displayed in the project tree. This can be useful for creating utility scripts containing a common set of functions shared between scripts. These scripts typically require no user interaction and usually do not contain a [process] section.

For a more in depth discussion on levels, see this <u>forum posting</u>.

# **Script Variables**

- Overview
  - <u>Usage</u>
  - Scope
- Fixed Variables
- <u>Custom Variables</u>
  - Examples
- Runtime Variables
- Macros
- Bugs
- FAQ (on variables)

### **Overview**

Variables are expressions that you can use to store a value that can be changed as needed.

These variables can be used to avoid fixed settings and keep the script language as dynamic as possible. This also makes your script become portable and work under any path or quickly change script settings when needed.

We can group variable into 3 categories, Fixed, Custom, and Runtime, each one having a specific sequence which we detail over the next chapter.

#### **Usage**

Variables are meant to be used inside your commands while your script is running.

They allow you to make changes at a global level. Changing the value of single variable can change the way how the whole project is built.

When a project starts to run - a list of variables is created using this specific order level

- 1. Fixed variables
- 2. Custom Variables
- 3. Runtime variables

If needed, we can overwrite a variable with a new value, meaning for example that a fixed variable can start with a value which can later be changed if we add a variable with the same name on our script, overriding the previous fixed value.

This is only valid while this script is running and these changes will be removed when running another script.

These explanations may seem complicated now but it's simple to understand once you follow each one of the chapters below. You don't have to worry much about this information until you decide to begin coding your own scripts in a more efficient way. In either case, it will be detailed here should you need it one day.

#### Scope

The [variables] section can be placed in 3 different files. Where a variable is defined will determine it's "scope" or availability.

- System variables defined inside WinBuilder.ini are available to all projects and scripts
- Project variables defined inside script.project are available to all scripts inside the current project
- Script variables defined inside the script that is running (also called "local variables") are valid until the script is concluded.

Variables may also be created or defined as the result of various commands that are executed, and many of them have optional parameters that may change the scope of a give variable - for example the <u>Set</u> or <u>AddVariables</u> commands can add variables to the script.project file (giving them a more permanent scope, or to the internal table that is active for the remainder of the current run (something between the project and script levels defined above).

### **Fixed Variables**

These variables are created by default just before a project start to run.

WinBuilder retrieves information found inside your Operating System and a few others definitions related to your project settings.

- Date related (output present date settings)
  - %day%
  - %Month%
  - %Year%
- User environment variables
  - %TempDir% path to the temporary folder
  - %ProgramFilesDir% path to localized "Program Files"
  - %!Username% extracts current logged User name
  - %UserProfile% contains the path to the User Profile folder
  - %WindowsDir% path to your Windows directory
  - %WindowsSystemDir% in most cases pointing to your windows system32 folder
  - %HostOS% string representing the OS of the host running WinBuilder
    - if majorVersion <= 4 then HostOS = NT
    - if majorVersion = 5 and minorVersion = 0 then HostOS = W2000
    - if majorVersion = 5 and minorVersion = 1 then HostOS = XP
    - if majorVersion = 5 and minorVersion = 2 then HostOS = W2003
    - if majorVersion = 6 and minorVersion = 0 then HostOS = Vista
    - if majorVersion = 6 and minorVersion = 1 then HostOS = Win7
- WinBuilder specific
  - %BaseDir% folder from where WinBuilder.exe is running
  - %tools% path where the common Tools folder can be found
  - %version% current version of WinBuilder
- Project specific
  - %SourceDir% path to use as source for your project
  - %TargetDir% work folder that will have the project build files

- %ISOfile% filename of the image to be created
  %ISOdir% path of %ISOfile%
  %ProjectDir% path to the respective project folder

### **Custom Variables**

These are additional variables that can be added by script authors. To create them, place a section called [variables] into your script and write your variables in INI style. Remember that since you are creating a variable, you need to use the percent sign (%) to make sure it is identified as a variable.

#### **Examples**

1. For this example we are creating two variables, available while the script is running, which can be used inside script commands.

```
[variables]
%workdir%=c:\myPath
%project_title%="My Boot Disk"
```

2. In this example, we will create one local variable, as well as a second section that when the script is run, will be added to the variables for the remainder of this run, as well as a third variable that will be added to the script.project as a project level variable for all future runs of this project.

```
[variables]
%workdir%=c:\myPath

[Global-variables]
%global_variable%="some value"

[process]
AddVariables, %ScriptFile%, Global-variables, GLOBAL
Set, %Proj-Variable%, "NewValue, PERMANENT
```

### **Runtime Variables**

In addition to the variables explicitly defined, and as mentioned above, additional variables may be created while the script is running. To specifically create new variables you can use these script commands:

- set
- addvariables

In addition to these commands, other commands also use variables to store or return the results of their processing. This means that whenever you use any command that uses a variable to store a result, a new variable may be created if it didn't exist before.

Some examples of commands that may do this are:

- StrFormat
- Retrieve
- IniRead
- RegRead

and even the If command (for some of it's various checks). Mostly these variables will only be local in <u>scope</u>, but it is always something to watch out for when creating variables to hold temporary results.

One last detail - the run command also uses a form of runtime variables to pass parameters to another section. These variables are of a special format, and use the hash character (#) followed by a single digit (1-9). You should always validate the values that are being passed in each of these variables to ensure that everything works as you expect. And if you need more than nine parameters, use the available commands to pack and access the parameters.

#### **Macros**

Macros are also defined in the [Variables] section. But, unlike Variables they do not have the percent signs (%). This is how the internal processing knows that they are **not** variables.

When used inside a script file, they can only be used at the begin of a line. They are used to substitute often used script command lines. Notice in the example below that if you are going to use any sort of parameter substitution, you would also need to create a separate section of code to be executed with the parameters being passed to that section.

For example, the code

```
[Process]
FileCopy,%source_sys%\aclui.dll,%target_sys%
FileCopy,%source_sys%\user32.dll,%target_sys%
...
FileCopy,%source_sys%\kernel.dll,%target_sys%
```

could be replaced by defining a macro to call a separate section of code that would execute the file copy command.

```
[variables]
Copy=Run,%ScriptFile%,DoCopy,%source_sys%,%target_sys%

[DoCopy]
FileCopy,#1\#3,#2

[Process]
Copy,aclui.dll
Copy,user32.dll
...
Copy,kernel.dll
```

It could also be simplified in this particular case, since we don't really need to pass in the directories. This means we could embed them directly in the secion of code to be processed

```
[variables]
Copy=Run,%ScriptFile%,DoCopy
```

```
[DoCopy]
FileCopy,%source_sys%\#1,%target_sys%

[Process]
Copy,aclui.dll
Copy,user32.dll
...
Copy,kernel.dll
```

Both sets of code would produce the same results!

### **Bugs**

There are a few known issues with variables that you should be aware of.

- Adding too many global or project variables on the list could start decreasing performance recommended to keep variables list on local scripts and projects as minimum as possible. **This also helps avoid unnecessary collisions in application scripts**
- If the value of a variable A depends on the value of variable B and you decide to change the value of variable B your variable A will not be updated and will still use the original value of variable B. If necessary, you can call <a href="mailto:system,RebuildVars">system,RebuildVars</a>.

Please let us know if this page is not complete or if you still haven't found needed information.

# **FAQ** (on variables)

Q: I'm still a bit confused, how can I understand this better?

A: Look inside a project folder and open up the script.project file with notepad to view good examples. This also applies to WinBuilder.ini (same folder as WinBuilder.exe) and to your script files.

Q: I'm doing my own script - how can I know if the value of my variables is right?

A: Use commands to output show the variables on the log. Echo,%myVar% is a good example to help in this case. Another good choice is the message command, since it can be made to wait until you click "OK" before processing continues.

Q: I know that we need to use the percent sign (%) to indicate something is a a variable, so it can **not** be used as a part of the variable name. Are there any other restrictions about what I can and cannot use?

A: You should only be careful to choose titles that aren't confused with other commands or variables, try using distinct titles that identify the variable purpose or use a unique prefex to ensure a project variable doesn't clash with any other scripts variables.

Q: Also heard that variables can replace commands, is this true?

A: Variables can't directly replace commands but can be used to run sections either within the current script, or any other available script. This is used to implement the <u>API</u> functionality. Also see the section on <u>Macros</u> above.

# **Graphical Interface**

- General description
- Component Types
- Component description
  - Added in WinBuilder 077 RC 2
- Example
- Known issues

# **General description**

The Interface section contains one ore more single lines describing the components shown in the WinBuilder window. It is driven from the contents of the [Interface] section of the script - UNLESS over-ridden by a setting in the [Main] section of the script.

The format of any individual line follows the basic form of any other INI file:

<name>=<content>

<name> is generated by WinBuilder when using the <a href="Interface Editor">Interface Editor</a> and contains the component type and an enumeration number. The <name> can be changed by the script author individually using the script editor if desired.

**<content>** contains one or more fields, separated by commas. The contents and number of fields will depend on the type of component being described. If any field contains a space, it is enclosed in quotes, or may be represented by the appropriate character sequence.

# **Component Types**

There are 15 different types, internally represented by a number between 0 and 14. They are:

- cmpTextBox = 0,
- cmpTextLabel = 1,
- cmpNumberBox = 2,
- cmpCheckBox = 3,
- cmpComboBox = 4,
- cmpImage = 5,
- cmpTextFile = 6,
- cmpEditValues = 7, ( **not implemented**)<sup>1</sup>
- cmpButton = 8,
- cmpCheckList = 9,
- cmpWebLabel = 10,
- cmpRadioButton = 11,
- cmpBevel = 12,
- cmpFileBox = 13,
- cmpRadioGroup = 14

<sup>&</sup>lt;sup>1</sup> This capability wasn't implemented due to a lack of perceived value. See <u>this</u> <u>post</u> for more information

# **Component description**

These first 7 fields are common to all implemented components:

- 1. A Text field that contains the following:
  - cmpTextBox: Caption
  - cmpTextLabel: Caption
  - o cmpNumberBox: <Component name>
  - cmpCheckBox: Caption
  - o cmpComboBox: <Choice>
  - o cmpImage: <File name>
  - o cmpTextFile: <File name>
  - cmpButton: Caption
  - cmpCheckList: Name of section in scriptfile that contains the list<sup>2</sup>
  - cmpWebLabel: Caption
  - o cmpRadioButton: Caption
  - o cmpBevel: <Component name>
  - o cmpFileBox: <File name> or <Directory name>
  - cmpRadioGroup: Caption
- 2. Component Visibility (1=shown, 0=hidden)
- 3. Type of component (see above)
- 4. x-coordinate of component upper left
- 5. y-coordinate of component upper left
- 6. width of the component
- 7. height of the component

Fields 8 and higher (if present) depend on the component type:

- cmpTextBox: <Value Entered in Text Box>
- cmpTextLabel: <Font size>, <Font weight>
- cmpNumberBox: <Specified Value>, <Min>, <Max>, <Increment>
- cmpCheckBox: <True/False>[, <Section to run>]
- cmpComboBox: <Caption 1>, ..., <Caption n>
- cmpButton: <Section to run>, <Picture>
- cmpWebLabel: <URL>
- cmpRadioButton: <True/False>[, <Section to run>]
- cmpFileBox: The word 'FILE' in the case of file, nothing in case of

directory

• cmpRadioGroup: <Caption 1>, ..., <Caption n>, <Selected index, starting with 0>

<sup>&</sup>lt;sup>2</sup> The cmpCheckList function will display whatever is listed in the particular section up to an "=" sign, which is then followed by a either a 1 or a 0 (checked/unchecked). This status value can be retrieved using the existing "INI" file capability of WinBuilder. See this post for more information and a sample "test" script.

### Added in WinBuilder 077 RC 2

Two additional (and optional) fields were added:

- 1. The cmpCheckBox, cmpRadioButton and cmpButton now support an optional parameter that will <Show progress window when running>
- 2. All components now support optional Tooltips. This field starts with a double underscore \_\_ as marker. When added, it will generate a "tool-tip" that will be displayed if the mouse hovers over the object.

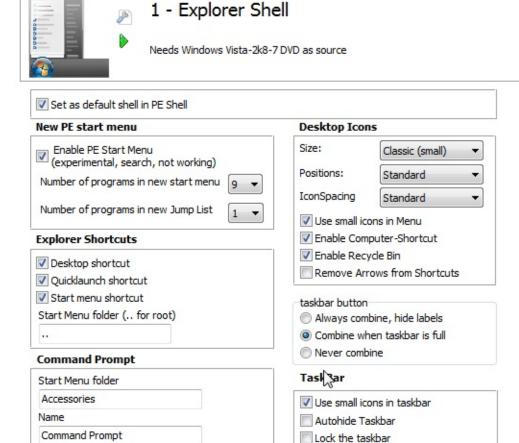
## **Example**

This example code has two pages of interface, and a button that toggles between them (i.e. causes code to be executed)

```
[Interface]
pCheckBox_PeShell="Set as default shell in PE Shell", 1, 3, 17, 11, 173, 2
pBevel1=pBevel1, 1, 12, 10, 6, 487, 32
pTextLabel10="New PE start menu", 1, 1, 17, 39, 114, 18, 8, Bold
pCheckBox StartMenu=" Enable PE Start Menu
                                                                  (experime
pTextLabel11="Number of programs in new start menu", 1, 1, 22, 96, 197, 18
pScrollBox11=9,1,4,217,95,36,21,5,6,7,8,9,10
pTextLabel12="Number of programs in new Jump List", 1, 1, 22, 125, 196, 18
pScrollBox12=1,1,4,217,125,37,21,1,2,3,4,5,6,7,8,9,10
pBevel10=pBevel10, 1, 12, 10, 54, 255, 99
pTextLabel20="Desktop Icons", 1, 1, 291, 39, 90, 18, 8, Bold
pTextLabel21=Size:,1,1,292,61,36,18,8,Normal
pScrollBox21="Classic (small)", 1, 4, 375, 60, 108, 21, Standard, "Classic (
pTextLabel22=Positions:,1,1,292,87,58,18,8,Normal
pScrollBox22=Standard, 1, 4, 375, 85, 108, 21, Standard, "Custom (in Source)
pTextLabel52=IconSpacing, 1, 1, 292, 111, 79, 18, 8, Normal
pScrollBox52="Standard", 1, 4, 375, 109, 108, 21, Minimum, Standard, Maximum
pCheckBox21="Use small icons in Menu", 1, 3, 292, 135, 148, 18, True
pCheckBox22="Enable Computer-Shortcut", 1, 3, 292, 153, 162, 18, True
pCheckBox23="Enable Recycle Bin", 1, 3, 292, 171, 122, 18, True
pCheckBox24="Remove Arrows from Shortcuts", 1, 3, 292, 189, 178, 18, False
pBevel20=pBevel20, 1, 12, 284, 54, 212, 157
ptaskbarbutton="taskbar button", 1, 14, 284, 222, 213, 73, "Always combine,
pTextLabel_SC="Explorer Shortcuts", 1, 1, 17, 156, 125, 20, 8, Bold
pCheckBox_DT="Desktop shortcut", 1, 3, 17, 179, 155, 18, True
pCheckBox_QL="Quicklaunch shortcut", 1, 3, 17, 197, 122, 18, True
pCheckBox_SM="Start menu shortcut", 1, 3, 17, 215, 122, 18, True
pTextBox_SMF="Start Menu folder (.. for root)", 1, 0, 20, 251, 138, 21, ...
pBevel30=pBevel30, 1, 12, 10, 172, 255, 105
pTextLabel50=TaskBar, 1, 1, 292, 300, 59, 18, 8, Bold
pCheckBox51="Use small icons in taskbar", 1, 3, 292, 324, 142, 18, True
pCheckBox52="Autohide Taskbar", 1, 3, 292, 343, 115, 18, False
pCheckBox53="Lock the taskbar", 1, 3, 292, 361, 109, 18, False
pCheckBox54="Disable Preview Desktop", 1, 3, 292, 379, 154, 18, False
pCheckBox55="Last active click selection in taskbar", 1, 3, 292, 397, 194
pCheckBox56="Show all tray icons", 1, 3, 292, 416, 120, 18, True
pTextLabel51=Taskbar, 1, 1, 292, 440, 47, 18, 8, Bold
pScrollBox51=Bottom, 1, 4, 348, 438, 81, 21, Bottom, Left, Right, Top
pBevel50=pBevel50, 1, 12, 284, 319, 214, 144
pTextLabel40="Command Prompt", 1, 1, 18, 281, 125, 18, 8, Bold
```

```
pTextBox41="Start Menu folder", 1, 0, 20, 319, 170, 21, Accessories
pTextBox42=Name, 1, 0, 20, 357, 170, 21, "Command Prompt"
pBevel40=pBevel40, 1, 12, 10, 296, 255, 88
Shutdown_Box="Enable ShutDown Button (x86 only)",1,3,17,403,210,18,F
pBevel60=pBevel60, 1, 12, 10, 396, 255, 31
pButton1="Additional Options", 1, 8, 10, 433, 120, 32, Additional Options, 0,
//
[Interface-Additional]
pTextLabel100="Explorer Additional Options ",1,1,12,3,182,20,9,Bold
pCheckBox101="enable Security-Tab", 1, 3, 17, 40, 170, 18, True
pCheckBox102="ISO Date-Time Format", 1, 3, 17, 58, 170, 18, True
pCheckBox103="Enable Previous version Tab ",1,3,17,76,170,18,True
pCheckBox104="Enable Sharing-tab", 1, 3, 17, 94, 170, 18, True
pCheckBox105="add File-Attributes Column", 1, 3, 17, 112, 170, 18, True
pCheckBox106="Add Encryption(* Beta)", 1, 3, 17, 130, 153, 18, False
pBevel100=pBevel100, 1, 12, 10, 36, 255, 116
pTextLabel110="Explorer Setting", 1, 1, 20, 160, 105, 18, 8, Bold
pCheckBox110="Show Menu bar", 1, 3, 17, 180, 104, 18, True
pCheckBox111="Show all folders", 1, 3, 17, 198, 104, 18, True
pCheckBox112="Automatically expand to current folder", 1, 3, 17, 216, 222
pCheckBox113="Add Recycle Bin to My Computer", 1, 3, 17, 234, 200, 18, True
pCheckBox114="Add Run to My Computer", 1, 3, 17, 252, 159, 17, True
pCheckBox115="Add Administrative Tools to My Computer", 1, 3, 17, 270, 23
pCheckBox116="Add Control Panel to My Computer", 1, 3, 17, 288, 200, 18, Tr
pBevel110=pBevel110, 1, 12, 10, 176, 254, 134
pTextLabel120="Start Menu Icons", 1, 1, 293, 20, 101, 18, 8, Bold
pCheckBox120="Show Run", 1, 3, 292, 40, 77, 18, True
pCheckBox121="Show My Computer", 1, 3, 292, 58, 117, 18, True
pCheckBox122="Show Control Panel", 1, 3, 292, 76, 119, 18, True
pCheckBox123="Show Admin Tools", 1, 3, 292, 94, 114, 18, True
pCheckBox124="Show NetPlaces", 1, 3, 292, 112, 99, 18, True
pCheckBox125="Show Favorites", 1, 3, 292, 130, 98, 18, True
pBevel20=pBevel120, 1, 12, 284, 36, 224, 116
pButton2="Main Options", 1, 8, 284, 176, 120, 32, MainOptions, 0, True, _MainO
//
[AdditionalOptions]
//Write or update an entry in the [main] section to set a new interf
IniWrite, %ScriptFile%, Main, Interface, Interface-Additional
System, REFRESHINTERFACE
[MainOptions]
IniWrite, %ScriptFile%, Main, Interface, Interface
System, REFRESHINTERFACE
```

This would result in the following two pages being displayed by WinBuilder



Enable ShutDown Button (x86 only)

Additional Options

Disable Preview Desktop

Show all tray icons

Taskbar Bottom

✓ Last active click selection in taskbar



#### 1 - Explorer Shell

Needs Windows Vista-2k8-7 DVD as source

#### **Explorer Additional Options**

- venable Security-Tab
- ✓ ISO Date-Time Format
- ▼ Enable Previous version Tab
- ▼ Enable Sharing-tab
- ✓ add File-Attributes Column
- Add Encryption(\* Beta)

#### Explorer Setting

- ✓ Show Menu bar
- ✓ Show all folders
- Automatically expand to current folder
- Add Recycle Bin to My Computer
- Add Run to My Computer
- Add Administrative Tools to My Computer
- Add Control Panel to My Computer

#### Start Menu Icons

- Show Run
- ✓ Show My Computer
- ✓ Show Control Panel
- ✓ Show Admin Tools
- ✓ Show NetPlaces
- ✓ Show Favorites

Main Options

## **Known issues**

If you change a script containing WB 077 features with WinBuilder 076 or lower, the features are lost!

## **Link Files**

The WinBuilder scripting environment provides two mechanisms to add "links" which allow it to reference external script files as a part of more than one project.

### At the Project File Level - a.k.a. folder.project

The first allows a link to an external folder to be specified. This is done by creating a file named folder.project in a projects folder. This file should contain a links section which specifies the external folder name containing the scripts to be imported.

All scripts in the specified folder and any sub-folders will be added to the folder containing the folder.project file. If any linked scripts are disabled then the script will also be disabled in the source project.

An example folder.project is:

```
[Links]
Projects\Gena-Complete\*.*
```

This links to another folder in the same projects directory (i.e. is is not an absolute path, but is relative the WinBuilder %BaseDir%. On the other hand, this example links to a completely specified path (which may be on another drive).

```
[Links]
D:\MyScripts\*.*
```

### At the Script File Level - a.k.a. <script>.link

The second mechanism uses a file with an extension of .link to link to a particular script file. If the script is disabled in the link file then this only affects the project containing the link file. In other words, the link file and the original script are independently able to be selected.

In other words, the original source script is left unchanged. The other settings are always stored in the original script, so if the settings are changed, it affects both the original and the linked project.

An example link file is:

[Main]
link=Projects\Addons\AccessGainDrivers.script
Selected=True

This links to another script in the same projects directory (i.e. is is not an absolute path, but is relative the WinBuilder %BaseDir%. On the other hand, this example file links to a completely specified path (which may be on another drive).

[Main]
link=C:\MyScripts\Addons\AccessGainDrivers.script
Selected=True

## **WinBuilder Tools**

- Overview
  - Compatibility
- Script Editor
  - Interface Editor
  - <u>Description</u>
  - Attachments
  - Script File Options
  - Source Code
- The Tools Tab
  - WinBuilder Options
  - Create new script
  - Convert Tool
  - <u>Upload</u>
  - License Agreement
  - Language Support
- Other Tools
  - Code Box
  - <u>Download Center</u>
  - WinBuilder .ini
- Frequently Asked Questions
- Bugs

## **Overview**

There are many tools that can be used to assist in the development of scripts or to help in the process of completing a project. In addition, there are also some tools available for specific tasks.

We can divide these Tools under two categories and will also describe in more detail the internal WinBuilder tools. These tools were added with the purpose of combining the several steps necessary to produce, test, run and publish new scripts.

Whenever this information is not complete enough and you have doubts on how to make things work, don't hesitate in placing your questions on our forum so that aditional information can be added to these pages. We also welcome requests on the forum for new tools to be added in the future.

## **Compatibility**

These tools were created with the intention to work under Windows 32 bit environments since Windows 95 up to Windows Vista. No major compatibility issues are present, except for limitations specific to individual tool. For example, tools like WimUtil should be executed on NTFS partitions to properly handle WIM images.

You might also be required to run some of these tools under an account with Administrative level privileges.

## **Script Editor**

Scripts are not static and you can edit or modify them as needed - this is a very simple and intuitive process. Select the script you wish to edit from the Main window and then click on the Edit button (the small icon that looks like a little wrench). This script editor contains several tabs - each one with a specific purpose



Note: Clicking on the logo image of the script will also enter edit mode.

#### **Interface Editor**

With this tool you can manage the objects placed within the interface of the script. These objects are very simple to handle. You just need to press the add button and select the object you wish to add from a list.

If you wish to remove a component you should select it and then press the delete button.

Selecting an object will also allow you to resize it and add your customizations. Some scripts have this interface locked - clicking on the small lock at the upper-right corner will unlock the interface.

You can also select multiple items. To do this, use the SHIFT button when selecting the various items you wish to adjust as a group.

## **Description**

On this tab you can write the details that will be mentioned on the main section of your scripts.

It also lets you select a logo image to help identify your script. You can use images in some popular formats like BMP, JPG and GIF - animated Gif's are also supported.

### **Attachments**

Use this tool to add files inside your scripts.

To allow your script to properly include binary files and still be editable with any text editor - these files will be encoded into an text form. These files can later be extracted while the script is running to the location you wish.

• When adding your files, please be careful not to include very large files or software that are not authorized to be freely distributed inside scripts.

## **Script File Options**

On this tab you can change the behavior of the script when it is enabled (i.e. Selected to be run) from the main window. It was added to ensure that other required scripts or even other scripts that are incompatible get disabled or enabled (i.e. selected or deselected) automatically.

In each box you can specify the path to the scripts you wish to control. To avoid using static paths you can use these variables to make the path dynamic

- %ProjectDir% is the full path to your project folder
- %ScriptDir% it's the path where this script is located
- %BaseDir% is the path where WinBuilder.exe was executed from (base)

You can also list more than one script on this text box, just split each script with a comma. Be careful of any spaces in the data you enter, which is why it is best to use the variables mentioned above to help isolate problems that might be caused by the location of the project.

- Dependencies If this script is "enabled" (i.e. goes from an un-selected state to selected, all the scripts listed here will be enabled (if possible), and all the files listed will be checked to make sure that they exist. **Notes:** 
  - 1. This means that if you select a script that has dependencies, WinBuilder will try to find those dependencies and if it is a script, will mark that script as "enabled" i.e. mark it with a checkbox so it will be executed.
  - 2. This functionality can only "enable" dependent scripts. It will **NOT** turn them off (or deselect them) if this script is later un-selected.
  - 3. The user could still decide **not** to run these scripts by manually "unselecting" them later.
- Disable Only scripts should be listed here. When this particular script is "enabled", WinBuilder will search for the listed script files, and try to disable them (i.e. make sure they are not currently "selected" to be executed. **Notes:** 
  - 1. This functionality can only "disable" scripts. It will **NOT** turn them back on (or select them) if this script is later un-selected.
  - 2. The user could still decide to run these scripts by manually "selecting"

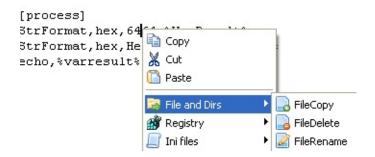
them later.

Finally, a note for script writers...If you change these settings, be sure to do a "refresh" of the scripts, since WinBuilder loads this information as it is parsing the scripts in the project - i.e. It is not dynamically reloaded as an individual script file is modified.

### **Source Code**

This tab allows you to edit the source code of your script. It's the place where you can view and add your commands.

One of the main benefits of using this editor is that there is a built in syntax help system. To use it, either use the "**magic wand**" icon, or just press the right-click (context) button of your mouse to view a menu that contains all available commands. Once you select a command it will be added in the place where your cursor is placed.



This command will also contain a quick template of the proper syntax as an example to show how to use it. With this menu you won't need to memorize all functions.

In the template that is inserted into the code, the items or values used in the command should be self-explanatory. In addition, the following conventions are used:

- [ ... ] means that this parameter or value is optional
- { ... | ... | ... ... } means that it requires only one of the items between the separators (|)

## **The Tools Tab**

There are many optional settings and tools available by selecting the tools function

### WinBuilder Options

The options tab allows to change some of the WinBuilder executable settings to work as you prefer.

#### Local Registry

This box contains settings that allow you to associate WinBuilder related files to open with your favorite text editor. You have the option to add the file association to your registry and later remove it if needed.

You can also make sure that WinBuilder assigns script files to be handled by your favorite text editor when it is started.

#### • Log Options

Control the behavior of your log window after a project is concluded.

- 1. The option "View in browser after saving" means that when you press the Save log button inside the log window your log.html file will be created and opened using you favorite browser.
- 2. The option "Open log window after process" will automatically show you the contents of the log window when your project has finished running all scripts. This is an useful option when debugging your projects and scripts.
- 3. The other options control the amount and types of information put into the log file. They are provided here as an easy way to change the settings stored in the <u>WinBuilder.ini Settings file</u>.

#### • Network support

You can enable or disable network support inside WinBuilder. This affects the following:

- Web tab
- Download Center
- FTP client
- New Download Server

## • Single script upload

Use this option under restricted environments where the program is not authorized to access the internet.

## **Create new script**

This is a tool that helps creating new scripts and use a small set of templates and wizards to make this process as user friendly as possible.

To create a new script you will only need to specify where you wish to place it using the "Location" box and write a filename under the "Filename" box.

On the right you find a text window that is already filled with some default values.

Pressing the create button will create a script with details and code you've written.

## **Convert Tool**

Use this tool to convert files from one file format into text compatible to be used inside scripts.

You can convert the following formats **Text files** BartPE plugins

Please note that the bartPE plugins does not completely translates all code and will only help converting most of the plugin into usable script format - you will still need to edit and adapt the resulting code as needed

### **Upload**

This is a very important tool for anyone who wishes to create a new Download Server. Imagine you are a project author who wants to make his project available via the WinBuilder download functionality.

#### What you have to do is:

- 1. In a WinBuilder project folder containing ONLY your project, click the "Tools button, then the navigate to the "Upload" tab.
- 2. Fill in the two text fields and click "Create".
- 3. Upload (preferred FTP) your complete project folder to your server.

The first field on this form is the name of the project as it will appear in the download tab, so you should probably use the name of the project. The second field is used to specify the name of a default web page that will be displayed next to the project folder when in the download view. Use this to give an overview of the project and to provide other links that might be useful.

• Note: You might just want to put the html file inside the project folder so it will already be uploaded to the web server.

When you click the "Create" button, WinBuilder will generate the following two files (updates.ini and index.html) in your project folder. The first is used by winbuilder to manage the update/download process. At this point, you have a complete set of files that can be accessed over the web through the WinBuilder download interface, including a very good HTML formatted "index" for the project which has the names, versions, sizes, descriptions and authors of all the script files that are part of the project.

## **License Agreement**

Clicking on the "View license Agreeement" box will open a new window that will display the terms of WinBuilder's license. As you might know, this program is freeware for both personal and commercial uses - but distribution and usage has to comply under the terms of this license.

It's not meant to be restrictive to the end user since WinBuilder is always provided for free in good faith, but these are the formal terms in which WinBuilder can legally be used and shared. These terms were needed to be written in order to avoid any abuse of rights in the future.

To avoid having to distribute WinBuilder.exe along with a separate license file these license terms were merged inside the binary itself where they are always available for everyone to read.

### Language Support

WinBuilder can support multiple languages and users can submit their own language files. Each term to be translated uses a numeric key to allow the same term be applied on similar cases. This avoids having to write thousands of messages which would only become a very tedious task for anyone proposing himself to complete a translation.

Since our program is constantly changing values and adding new features - languages would grow outdated quite quickly and surely most translators wouldn't always be available to provide the needed updates.

Whenever a language file is available, this program will compare it against the internal language database (in default english) and will add the missing terms.

This also means that even old translated language files that are not completed with the latest changes will still be valid to be used and you'll be able to complete yourself what needs to be corrected.

*How to create your own language wb?* - You can start by clicking on this button from the Language tab



This button will open a box asking for a filename to describe your language. Use only english compatible characters and description so that it can be understood under as many languages as possible.

**How to edit your language?** - On the same tab you find the window that lists all values and corresponding translated terms - just write each term the way it best fits your language and press the save button.

Where do I find my language file? - All language files are placed under %BaseDir%\Projects\Tools\Languages (remember that %BaseDir% is the same path from where you run WinBuilder). If this folder doesn't exist, English language is used as standard - when creating a new translation file

this folder is automatically generated. You can edit any of these language files with a simple text editor.

**How do I share my language file?** - Look on community forums for the language specific sub-forums where you might find more people that share the same language, if there isn't a forum created then you can post on the main WinBuilder discussion forum.

## **Other Tools**

#### **Code Box**

Code box is a tool added to quickly help you test new commands. It is an *Advanced* command and must first be enabled from the Options page.

It will create a small template script that will be executed whenever you press the play button inside this tab. With this tool you can run commands to see how they work and view the results on the log window.

You can also add commands using the right click menu in the same way as the internal source code editor. Each code box script runs exactly under the same conditions as any other script inside your selected project, making this a good tool for debugging variables and commands.

The code you add inside the Code Box is automatically saved inside a file called CodeBox.txt which can be found on the root of each project where you use it - this way we ensure that you can have a Code Box specific to a single project.

#### **Download Center**

The Download interface was added to help people keep in touch with the community. From this page you can quickly view the latest discussed topics from our forums.

It also may allow you (if supported by the project) to update your scripts without having to download the entire project. Scripts may be marked with specific "Download level" in the [MAIN] section of the script file. This allows you to focus on checking the project according to the level specified.

The intention is that the script writer or project owner would mark scripts starting from the most important/critical to the least ones. So:

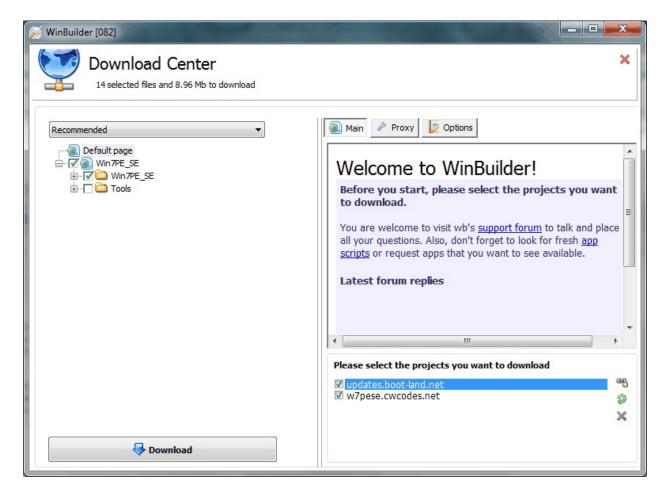
- Level 1 stands for core scripts that the project cannot be built without.
- Level 2 includes the core scripts and scripts that add a normal environment without much fluff
- Level 3 adds up all the scripts in the previous levels and scripts on this level add fluff and special features
- Level 4 is intended to specifically mark scripts available as beta to interested testers, was never used by developers. I am not sure if it will includes all scripts marked in previous levels.

These levels are displayed as

- Minimum (i.e. Download\_level=1)
- Recomended (i.e. Download\_level=2)
- Complete (i.e. Download\_level=3)
- Beta (i.e. Download\_level=4)

The script on the project server will be process if the value selected is greater than or equal to the value included in the script. This means that if the value is equal to zero (or is not specified), the script will always be processed.

Scripts are compared based on the version number in the script's [MAIN] section by default.



This check can be changed as one of the options (in the options tab) on this screen. In addition to the Options tab, the different servers that are checked can also be managed on this screen as well. These are stored in the WinBuilder INI file, but the graphical controls on the lower right side of the screen are provided to allow you to add new links, cause WinBuilder to refresh the script checks with the selected servers or to remove the selected links.

## WinBuilder.ini

The WinBuilder.ini settings are documented <a href="here">here</a>

## **Frequently Asked Questions**

Q: I want to create a new script with these tools, but I still have lots of doubts about scripting and all things that need to be done to get my programs working - where can I find more info?

A: Perhaps the best help you can get is reading other scripts that work the way you need and apply them to run your programs. Don't forget to also ask people whenever you find something a bit harder to understand or do, nobody knows everything from scratch so take one step at time and things will start to make sense quickly.

## **Bugs**

Bugs and requests are welcome on our discussion forum dedicated to these reports here.

Please remember to add as much details as possible and it would also help if you described exactly how this bug can be reproduced.

Also check if you're also using the latest version since a lot of bugs tend to be corrected as new versions become available.

Whenever in doubt you can also help yourself and use the Search function inside the forums to find other members who might have reported similar cases to yours. You might even find answers to how they've solved the same issue or add your comment to ensure that it gets solved sooner.

## WinBuilder and Script FAQ

## Where do I place my scripts?

You can place your scripts inside the same folder where the project you want to create is placed. Using LiveXP as example, you can place your scripts inside the <code>%BaseDir%\Projects\LiveXP</code> folder, where <code>%BaseDir%</code> is a variable used by WinBuilder to indicate the same path where <code>WinBuilder.exe</code> is found.

Placing your script inside this folder is more than enough to make it available from the the main window when browsing your project. Also worth noting that some projects also have a specific folder where related scripts are grouped. Using the same LiveXP example, you should place your scripts that deal with "Applications" inside the %BaseDir%\Projects\LiveXP\Apps folder.

# Why is it that some scripts only work with some projects?

Even thought all scripts are created equal and use the same scripting language - different projects can use very different ways to add programs and functions.

In most cases you should understand which projects are compatible with your script before trying to use it. There is also an ongoing effort to make scripts work in a wider variety of projects, read here for more details.

If you are using the Download Center as source to get your scripts from the internet, then you won't need to worry about this since each server only contains scripts that are always correctly placed inside compatible projects.

If a script is growing outdated, you can try to adapt it yourself. Pick a similar script that works the way you need and use it as example, adapting things as needed.

You can also post all your inquiries about .script file coding in our community where others can tell you which specific steps may be needed or even teach how to update your script.

### How do I easily edit or create a script?

It's recommended to use WinBuilder's internal editor to let help you customize the script and highlight the source code syntax while scripting.

You can use WinBuilder's internal editor by selecting a script and clicking on the Edit button.

When running WinBuilder, click on the Tools button where you can also find a small utility to help you creating your scripts from available templates.

#### Note:

- The learning curve to make your own script is very simple and acessible.
- You can use other scripts as example to create your own scripts and then share them in the forums to help others if you wish also a good place to ask for scripts that support the programs you like.

# How can I use WinBuilder from the command line? (for example to automate a build)

In addition to the <u>existing documentation on the command line options</u>, see this posting over in the forums: <u>How To use WinBuilder command line</u>

### WinBuilder.ini

Certain aspects of WinBuilder's exe are stored or can be controlled using variables and values found in the WinBuilder.ini file. These settings may appear in any order, but are grouped below based on their functionality. Unless otherwise noted, **ALL** settings are store in the [MAIN] section of the INI file.

- General Appearance / Information
- Network Settings
- Download Center
- General Behaviour
- Register Extensions
- Log File Options
- Custom
- Syntax Highlighting
  - Formatting
  - Font Style
  - Colors
  - Syntax Groups
  - Defaults

# **General Appearance / Information**

X	Stores the <i>Left side</i> coordinate used to create the WinBuilder window. It is updated when exiting WinBuilder	
XX	Stores the <i>Width</i> used to create the WinBuilder window. It is updated when exiting WinBuilder	
Y	Stores the <i>Top</i> coordinates used to create the WinBuilder window. It is updated when exiting WinBuilder	
YY	Stores the <i>Height</i> used to create the WinBuilder window. It is updated when exiting WinBuilder	
SP	Sets the <i>Width</i> of the script tree pane inside the Main Window. It is updated when exiting WinBuilder	
InterfaceGrid	Sets the number of pixels used for the Grid inside the interface editor. Set this to <i>0</i> if no grid is desired	
Last_Project	Name of the <i>last project</i> used by WinBuilder. It is updated when exiting WinBuilder	
Enable_CodeBox	Set this to <i>True</i> to show the "CodeBox" option on the main window. Can also be managed using the Advanced setting on the "Tools" Tab	
Enable_RefreshButton	Set this to <i>True</i> to show the "Refresh" option on the main window. Can also be managed using the Advanced setting on the <u>"Tools"</u> Tab	
Language	Language of the string translation file(%Tools%\Language\????.txt) See also <a href="LanguageSupport">Language Support</a>	
FileVersion	Contains the version of the WinBuilder.exe last used	
RunExternal	HAVE NO IDEA	

### **Network Settings**

The following are global network settings.

DisableNetwork	If set to <b>1</b> will disable the network, which switches the download functionality generally off. When set to <b>0</b> or not supplied, the network is enabled	
WebCheckDomain	Winbuilder will query this domain to determine if web access is available.	
AskForDownload	Ask user before downloading any file requested by scripts via the WebGet or WebGetIfNotExist commands.	
UseTimeout	If AskForDownload is specified continue with download after this timeout period.	
ValueTimeout	Sets the <i>Timeout</i> value in seconds.	

The following settings are stored under the <code>[Proxy]</code> section. As the login information is encrypted you should only configure these setting from inside the winbuilder download center.

useHTTPProxy	Use web proxy (True/False)	
HTTPhost	Host address for web proxy	
HTTPport	Port number of web proxy	
HTTPuser	Encrypted User Name for web proxy	
HTTPpwd	Encrypted Password for web proxy	
HTTPRFC2617	Use Basic Authentication for web proxy (True/False)	
useFTPProxy	Use FTP proxy (True/False)	
FTPhost	Host address for FTP proxy	
FTPport	Port number of FTP proxy	
FTPuser	Encrypted User Name for FTP proxy	
FTPpwd	Encrypted Password for FTP proxy	

FTPRFC2617	Use Basic Authentication for FTP proxy (True/False)	
DNS	DNS Server	
PingType	Ping Type (0=ICMP 1=HTTP - use if your firewall blocks ICMP)	

### **Download Center**

These settings control the web sites listed in the download center. See <u>this page</u> for more information, including how to use the included interface to manage the list of servers being used The following are used to control download center preferences.

WebServer	A comma separated list of all the servers you may check for updates.	
WebServer_Selected	A list of the servers actually selected for download processing	
Last_Download_Level	Last level used in download center ( <b>0</b> =minimum <b>1</b> = recommended <b>2</b> =complete <b>3</b> =beta)	
DN	Sets the <b>Width</b> of the script tree in Download Window	
Allow_Beta	Allow use of the <b>beta</b> level download (i.e. level 3)	
AutoDownloadTree	Rebuild the download tree only when the user clicks on the green arrow (i.e. a value of <b>True</b> makes green arrow visible)	
StopDownloadTree	Allows user to interupt a build of the download tree (i.e. a value of <b>1</b> makes the red stop square visible)	
RenameCurrent	If set to <b>True</b> , WinBuilder will rename an existing (i.e. current) script file rather than overwrite it. If set to <b>False</b> (or not specified), WinBuilder will overwrite any existing script when downloading new content	

### **General Behaviour**

ErrorStop	If set to <i>True</i> , WinBuilder will stop the build when an error occurs.	
ExtractWarn	During script processing, warn when extracting an attached file will overwrite an existing one	
OneCore	When WinBuilder runs on a multicore system, use only one core. Can also be managed using the Advanced setting on the "Tools" Tab	
ParameterCount	When processing, check script command parameter counts.	
SourceIndenting	Indent source code for better readabilty	
!Autosave	Autosave edited interface / souce code (0=On 1=Off 2=Ask)	

# **Register Extensions**

RegisterExtensionsWith	Register WinBuilder extentions (i.e. *.script) to an external editor.
RegisterExtensionsDouble	Set the <i>Double click</i> syntax
RegisterExtensionsRight	Set the <i>Right click</i> syntax

# **Log File Options**

ForceWarning	Switches "Debug Log" on / off	
VerboseWarning	Shows more warnings in Debug Log	
LastLog	Path of active log.html	
LogAuto	Show log after build, even if it is error-free	
LogComment	Show / suppress comments in source code	
LogEnglish	Generate English log file, independent of String translation language	
LogOriginal	Add original script lines to log	
LogStop	Generate log.html when user clicked the blue "Stop" button	
LogTimeStamp	Add time stamp to file name generated - otherwise, file is just named "log.html"	
!noIconLog	Put strings (i.e. "[Success]" etc.) into logs rather than using pointers to icons	
!noTimeOutput	Do not output time in log (This can make it easier to compare two different log files)	
SyntaxWarn	Produce a <i>Warning</i> on unresolved (misspelled) commands	
ShowVariables	Add a list of variables to top of log - default is "True"	
ViewLogInBrowser	Automatically show log in browser after generating log.html	

### **Custom**

These settings are stored in the [Custom] section of the .ini file and allow you to "Brand" Winbuilder for your project.

Title	This text will be added to the Winbuilder title bar
Logo	Path to an image file (gif/jpg/bmp) that will be used in place of the Winbuilder logo on the main screen.

### **Syntax Highlighting**

You can configure the syntax highlighting of winbuilders internal code editor to a user defined color theme. At this time only styles and colors may be changed. You can not define commands or categories to be highlighted.

You may configure the following properties:

- Font Style (font: bold, italic, strike-through, underline)
- Foreground (color of the text)
- Background (color of the background)

#### **Formatting**

Configuration is defined by a series of keywords seperated by the pipe | operator.

Foreground:<color>|Background:<color>|Style:<font style>

#### **Font Style**

Value	Description
fsItalic	Italics
fsBold	Bold
fsUnderline	Underline
fsStrikeOut	StrikeOut

You may specify multiple font styles by using a bracketed comma separated list.

Example: Style:[fsBold,fsItalic]

#### **Colors**

Both Foreground and Background color can be configured using the following color table.

Value	Description	Value	Description
clAqua	Aqua	clBlack	Black
clBlue	Blue	clCream	Cream
clDkGray	Dark Gray	clFuchsia	Fuchsia
clGray	Gray	clGreen	Green
clLime	Lime	clLtGray	Light Gray
clMaroon	Maroon	clMedGray	Medium Gray
clMoneyGreen	Money Green	clNavy	Navy
clOlive	Olive	clPurple	Purple
clRed	Red	clSilver	Silver
clSkyBlue	Sky Blue	clTeal	Teal
clWhite	White	clYellow	Yellow

#### **Example 1** - Only Define Foreground Color:

Foreground:clMaroon

### **Example 2 -** Define both Foreground and Background color:

Foreground:clMaroon|Background:clGray

### **Syntax Groups**

Syntax highlighting is separated into groups, with each group having its own style. The following groups are defined internally by winbuilder. Currently you can not change these groups or define new ones.

Comment	Script comments starting with // or ##
String	Strings enclosed in quotes
Variable	script Variables
Number	Numbers

CommandItems	Internal commands
IfCommandItems	Conditional commands
StrFormatItems	String Format sub commands
SystemCommandItems	System commands
OtherItems	Other command items
BooleanItems	Boolean Operatiors
Params	Command parameters
Escapes	Escape characters

#### **Defaults**

[HighLighter]

Comment=Style:[fsItalic]|Foreground:clGray

String=Foreground:clRed Variable=Foreground:clGreen Number=ForeGround:clBlue CommandItems=Style:[fsBold]

IfCommandItems=Foreground:clMaroon|Style:[fsBold]
StrFormatItems=Foreground:clMaroon|Style:[fsItalic]

SystemCommandItems=Foreground:clTeal|Style:[fsBold,fsItalic]

OtherItems=Foreground:clTeal

BooleanItems=Foreground:clNavy|Style:[fsBold]

Params=Foreground:clTeal Escapes=Foreground:clTeal

# **Command Line Options:**

The following command line options can be used to automate the building of one or more projects.

Option	Description
/run=	Path to the project dir. (Ex. C:\WB\Projects\Win7PE)
/title=	Project Title
/x=	Window width
/y=	Window Height
/target=	Target Directory
/ISO=	Path to ISO that will be created
/log=	Path to log file that will be saved
/source=	Path to the source used for building
/showlog	Display the log when building is finished
/debug	Display a message box with project, target, source and ISO paths
/beep	Beep when building is finished

# **Examples:**

#### Example 1

Simple Command Line Usage to build a project.

#### Example 2

This **advanced** example uses a batch script to set some options and build the project.

```
REM ****** The next 4 lines must be defined individually
SET PriName=nativeEx_barebone
SET BaseDir=C:\Dokumente und Einstellungen\Peter\Desktop\WB Archiv\E
SET setOpt=%BaseDir%\pscProqs\setOptions\setOptions.exe
SET WB=%BaseDir%\WinBuilder071B1.exe
REM ****** The next 5 lines can remain unchanged
SET Project=%BaseDir%\Projects\%PrjName%
SET WBConst=/run="%Project%" /title="My Project" /x=10 /y=50
SET tg=%BaseDir%\Target\%PriName%
SET iso=%BaseDir%\ISO\%PrjName%
SET log=%BaseDir%\Support\Log\%PrjName%
REM ****** If the directory for your log files does not exist, you
REM ****** WinBuilder does not (yet) create it.
MD "%BaseDir%\Support\Log"
MD "%BaseDir%\Support\Log\%PrjName%"
REM ****** The next lines call Main with my individual OS / languag
REM ****** Maybe four you different statements are better suited
REM ****** Last line MUST be: "GOTO :EOF"
FOR /D %%s IN (V:\BCD\cds\XPP_*SP2) DO CALL :Main %%~ns %%s
FOR /D %%s IN (V:\BCD\cds\W2003_*) DO CALL :Main %%~ns %%s
GOTO : EOF
:Main
REM ****** Here you can build different sets of your project to pro
```

```
In the first set you have to define everything what is i
REM ****** and may be different from the actual project
REM ****** In the following sets you only need to define what is ch
REM ****** The setOpt program and this sample batch can be download
REM ****** http://nativeex.boot.../setOptions.zip
REM ****** index must be individual for each set and should contain
SET index=_%1_STD_R_R
REM Standard ISO, RAMDisk
"%setOpt%" "-0:!Fundamentals.script:Interface.pScrollBox1=B:" "-P:%P
"%setOpt%" "-0:!Fundamentals.script:Interface.pScrollBox3=RAM Drive"
"%setOpt%" "-0:RAMDisk.script:Main.Selected=True" "-P:%Project%"
"%setOpt%" "-0:CreateISO.script:Main.Selected=True" "-P:%Project%"
"%setOpt%" "-0:BootSDI.script:Main.Selected=False" "-P:%Project%"
"%WB%" %WBConst% /target="%tg%%index%" /ISO="%iso%%index%.iso" /log=
SET index= %1 SDI R R
REM BootSDI, RAMDisk, SettingsDrive = RAM Drive
"%setOpt%" "-0:!Fundamentals.script:Interface.pScrollBox3=RAM Drive"
"%setOpt%" "-0:CreateISO.script:Main.Selected=False" "-P:%Project%"
"%setOpt%" "-0:BootSDI.script:Main.Selected=True" "-P:%Project%"
"%WB%" %WBConst% /target="%tg%%index%" /ISO="%iso%%index%.iso" /log=
SET index=_%1_SDI_R_S
REM BootSDI, RAMDisk, SettingsDrive = System Drive
"%setOpt%" "-0:!Fundamentals.script:Interface.pScrollBox3=System Dri
"%WB%" %WBConst% /target="%tg%%index%" /ISO="%iso%%index%.iso" /log=
SET index=_%1_SDI_N_N
REM BootSDI, no RAMDisk
"%setOpt%" "-0:!Fundamentals.script:Interface.pScrollBox1=--" "-P:%F
"%setOpt%" "-0:RAMDisk.script:Main.Selected=False" "-P:%Project%"
```

"%WB%" %WBConst% /target="%tg%%index%" /ISO="%iso%%index%.iso" /log=

# **Compatibility**

- WinBuilder.exe is supported on Most modern windows operating systems. It may use Internet Explorer to access help pages and use the Web tab.
- It won't require any .NET framework or any similar component. It is mostly a standalone application without special external dependencies. (you can even disable network access).
- WinBuilder itself can run quite happily both on FAT32 or NTFS partitions BUT certain projects require a NFTS filesystem. Be sure to check the requirements of the project that you are using!
- On Vista and later versions, it will be necessary (in most cases) to disable UAC (User Account Control) to allow creation of the needed registry hives when building a new boot disk.

## **Known issues with Winbuilder.exe**

This is a list of issues/quirks/oddities with Winbuilder.exe that cannot be fixed or are difficult to fix at the present time.

Please don't post bug reports about these items.

# **Known issues:**

**Issue:** The code editor has a column limit of 1024 characters. [Bug Report]

**Workaround:** the Winbuilder engine itself is quite happy to read beyond the limit when processing the script so processing is not affected. One must just use another text editor such as notepad to modify the script.

**Remarks:** This limit will be raised to 10240 in the next release (WB83-WB84)

**Issue:** Bug with <u>GetParam</u> [Forum Topic]

**Workaround:** Use string format

[process]
Run,%ScriptFile%,Pack,1,2,something with space,4,5,6,7,8,9,10,11,12,
[pack]
PackParam,1,%packed%
GetParam,3,%Value%
Echo,%Value%
StrFormat,SPLIT,%packed%,#\$c,3,%Value%
Echo,%Value%

**Remarks:** Fixed in a future WB release (WB83-WB84)

**Issue:** Sometimes text labels do not show up or check boxes disappear when using themes other than "Windows Classic" [Development Topic]

**Workaround:** Use the windows classic them or disable Visual Styles in the Compatibility options for Winbuilder.exe

**Remarks:** This is a known issue with the current Delphi 7 compiler.

**Issue:** When using <u>RegImport</u> reg binary values are written incorrectly. [<u>Bug Report</u>]

**Workaround:** Convert the .reg file to WB script using <u>Reg2Wbs</u> or <u>RegCPE</u>. You can also use shellexecute to merge the .reg into the hive Shellexecute, hide, regedit.exe, "/s #\$qMyRegFile.reg#\$q"

**Remarks:** This is an issue with the Delphi7 function TRegFile.LoadFromFile(<filepath>); and cannot be corrected without updating the compiler to a newer version (\$\$\$).

# **Troubleshooting**

When a command doesn't work as expected you should follow these steps to ensure that it is a true bug and not a human error:

- Check the source code and see if the syntax is correct
- Add small tests to the used variables to check if the respective values are outputted correctly. (ex. echo,%myVariable%)
- Read the special conditions under which the project has to be run (found the project page)
- Check if there aren't any external interferences as locked files or no disk space on the target folder
- Try whenever possible from a fresh project to avoid any chance of corrupted files or invalid project settings

If none of the above solves your issue then here's the recommended way to proceed:

- Visit the community forums and search for similar reports of the same behavior as you notice
- If a similar case is found you should post a reply on this topic, otherwise start a new topic in the appropriate category
- Writing advice before posting a bug: Include a descriptive title in your message *including your log.html* file and a test script so that it can be debugged and used by others.
  - If the file is large, consider using a service such as MediaFire and post a link to where the log file may be downloaded from.
- Projects also follow their own script rules and these methods can change occasionally, whenever in doubt you should post your questions in each project forum discussion.
- Check the forums Changes in script language can be followed for each newer WinBuilder as changes to the script code are suggested and debugged by other members.

#### **AntiVirus Issues**

It was occasionally reported by a few users that some AntiVirus can interfere while scripts are copying or deleting files. It's advisable to temporarily disable the anti-virus **only when debugging** the building process to check for possible causes that keep your project from being concluded as expected.

**Under normal circumstances there is no need to disable your Antivirus.** But, please be aware that antivirus programs will scan any files being copied which may impede performance.

# **Reporting Bugs**

WinBuilder is a tool in progress and there are always bugs being reported and properly solved, we always appreciate your feedback and reports in case a bug is found.

Bugs can be divided in two types:

- 1. Core WinBuilder Visit the forum for <u>Winbuilder</u> and check if you're using the latest version. It's always good to use the latest versions since they contain several improvements at each time a new version is released. If you still have an issue, please enter a posting detailing the issue.
- 2. Project related Each project also has a specific forum where all related issues and fixes can be found, please visit the respective forum page of your project and post details about the issue there.

In both cases, when reporting for a bug it is recommended that your search the forums to check if any other person has already posted a similar report and post a comment in the respective thread. Your feedback will help others who may be able to assist in quickly finding a solution.

### License

This program is freeware and can only be used under the terms of the respective license.

The full scope of WinBuilder's license is available to read inside the program by clicking on:

Tools --> License

and is applied only to WinBuilder.exe.

Projects, Scripts and other files distributed along with WinBuilder are subject to their own licensing terms whenever applied and always according to the respective author terms - which should always be credited for their work when creating distributions or derivative versions of WinBuilder projects or scripts.

WinBuilder.exe is distributed on an "AS IS" basis, WITHOUT WARRANTY OF ANY KIND, either express or implied.

The Developer of WinBuilder.exe is Nuno Brito (mail@nunobrito.eu).

Copyright (c) 2006-2011 Nuno Brito. All Rights Reserved. All relevant trademarks are acknowledged.

# **Version History**

- WinBuilder Releases
  - [082] stable (2011-06-15)
  - [081] Beta 1 (2011-01-04)
  - [080] update (2010-03-03)
  - [080] update (2010-02-23)
  - [080] update (2010-02-20)
  - [080] (2010-01-15)
  - [078 SP6] (2010-01-03)
  - [078 SP5] (2009-11-23)
  - [078 SP4] (2009-11-02)
  - [078 SP3] (2009-09-25)
  - [078 SP2] (2009-09-06)
  - [078 SP1] (2009-09-02)
  - [077 RC 3] (2009-07-14)
  - [077 RC 2] (2009-06-10)
  - [077 RC 1] (2009-06-03)
  - [077 beta 3 hotfix] (2009-04-26)
  - [077 beta 3] (2009-04-26)
  - [077 beta 2] (2009-04-01)
  - [077 beta 1] (2009-02-05)
  - [076] hotfix (2009-02-02)
  - [076] (2009-02-02)
  - [075] beta 7 (2009-01-14)
  - [075] beta 6 (2008-09-18)
  - [075] beta 5 (2008-07-07)
  - [075] beta 4 (2008-06-10)
  - [075 beta 3] (2008-06-09)
  - [075 beta 2] (2008-05-13)
  - [075] (2007-12-18)
  - o [073] (2007-10-14)
  - o [071] (2007-05-14)
  - [069] (release date unknown)
  - [068] (2007-01-30)
  - [067] (2007-01-28)

- [052] (2006-10-07)
- [051] (2006-09-22)
- [050] (2006-08-21)
- [049] (2006-08-17)
- OpenBuilder Releases
  - o [048] (2006-07-27)
  - [047] (2006-07-18)
  - [046] (2006-07-15)
  - [045] (2006-07-11)
  - [044] (2006-07-08)
  - [043] (2006-07-04)
  - o [042] (2006-06-26)
  - [041] (2006-06-23)
  - [040] (2006-06-22)
  - [039] (2006-06-18)
  - o [038] (2006-06-18)
  - [037] (2006-06-16)
  - [036] (2006-06-07)

### WinBuilder Releases

### [082] stable (2011-06-15)

- changed comp80 default from ON to OFF
- fixed bug in interface editor with multiple selected components
- fixed bug in interface editor with weblabel in panel
- fixed lost custom title

#### published as release preview

- fixed bug in If command, when a compare string starts with "run"
- changed removed "orphaned" System, HasDOS8\_3 command
- fixed bug which in interface editor with multiple selections sometimes "forgot" single selections
- fixed bug introduced by removing of System, HasDOS8\_3 command
- added when image does not contain URL, click into the image starts the default graphical editor
- changed place weblabels in front of textlabels
- fixed [http://reboot.pro/14...post\_\_p\_\_131528 script editor bug with source code button (the light bulb)]
- fixed single run now displays script's description rather than script.project description
- fixed bug with endless loop in exiting WinBuilder, when a script has been deleted externally
- fixed When in interface editor activating "resize to fit" on an image, the corner dots of the image are not repositioned correctly until the image is clicked
- changed set "clear" button in interface editor / image to invisible
- added RefreshControl method to TInterfaceComponentManager. That hopefully avoids interface troubles with con-classic themes

frozen for second release preview as 82.0.0.15 on 2011-JUL-01

### [081] Beta 1 (2011-01-04)

- added System, HasUAC and System, Is Terminal commands
- fixed bug "RegMulti, HKLM, Section, Key, DELETE, SubVal bug"
- fixed bug "RegWrite, HKLM, 0x3, <Section>, <Keyname>, %variable%"
- fixed access violation in Set,%InterfaceVar%
- fixed bug "AddInterface does not work when used with run"
- changed "Add a new object" window in Interface Editor is sized depending on DPI
- added Please add the function to doublecklick a line the the combobox of "Add a new object window" to add the object
- added AutoSave mechanism to interface editor
- added When clicking the green 'run' arrow in source editor, it is (depending on AutoSave option) also asked for 'save' before starting
- added Interface editor now can group elements to adjust or move them all simultaneously
- added #\$z to magic wand
- fixed "Coolest Bug ever!" corrupting .scr file
- added AutoSave mechanism to description editor
- fixed License window sizing issue
- fixed Comma issue in If commands
- added Deleting all selected components in interface editor
- added Equal horizontal or vertical spacing of components in interface editor
- added Horizontal or vertical centering of components in interface editor
- added optional [Text] output in log rather than download icons.
- added optional supressing time output in log, in order to compare easier.
- added NOWARN parameter to FileDelete command
- fixed Bug in RegMulti, which overwrote %variables%
- fixed Bug in logging error of DirCopy command (wrote " + s" rather than error description)
- added Verbose option to Debug Log
- added NOREC option to FileDelete
- changed the string "#<number>" can be passed and is not seen as parameter
- fixed Codebox in a fresh started WB was without API
- fixed Bug in logging Loop statements

- changed certification logic
- added System, HasDOS8\_3 command
- fixed Bug in RegWrite,HKLM,0x1,WB-Setup\Classes\SHCmdFile\shell\open\ddeexec,," [ShellFile(""%1""#\$c""%1"",%S)]"
- fixed Bug in StrFormat,Rtrim
- changed %variables% as macros temporarily tolerated with log warning. In stable version this will be an error.
- fixed Parameter existing only of 2 quotes now becomes empty
- fixed Do not write surrounding quotes into variables when reading the [variables] section
- fixed resolve escapes to character in If command compares
- fixed (Finally) Do not write surrounding quotes into variables when reading the [variables] section
- fixed resolve escapes to character in If,QUESTION
- fixed several 'surrounding quotes' bugs in If command
- added menu entry in magic wand: copy complete command list to clipboard
- rewrote If command
- fixed log output for If commands
- fixed Bug in PackParam
- fixed Bug in RegWriteBin
- changed fine tuning of RegRead and RegReadBin commands
- fixed Incompatibility of GetParam, introduced by the bug fix of PackParam
- addded System,LOG,OFF / System,LOG,ON commands
- fixed Bug in StrFormat when writing result to parameter #n
- fixed Bug in log.html generation causing 'Index out of bounds' when logging a Loop command
- fixed Bug causing 'Access Violation' in interface
- changed Errors supressed by System, ERROROFF are now shown in Debug Log as 'Handled Errors'
- added icons for log.html can be optionally in %BaseDir%\Pictures, bypasses download
- fixed Bug in interface due to values with quotes
- fixed Bug in running section on CheckBox click
- fixed Bug in Interface Click event handling
- fixed Bug in If command, when comparing numerical values
- changed some cosmetics in output of log.html

- fixed Bug in OUT: parameters handling
- fixed Bug which did not allow GLOBAL changes of %Constants%
- fixed Bug with duplicating quotes in interface text labels
- fixed Bug in IniDelete
- changed variables replacement: from linear search to binary search
- fixed #\$xs sometimes not resolved
- improved log output
- fixed some smaller bugs when treating multiple components in interface editor
- fixed bug in progress window when exec a script at tree end
- fixed made behaviour of interface component 'FileCheckList' logically compatible to WinBuilder 076
- fixed bug when aborting "Add component to interface"
- fixed bug: Set,%InterfaceComponent% PERMANENT sometimes corrupts interface
- changed now If, EXISTFILE accepts wild cards?
- changed now If, EXISTDIR accepts wild cards?
- fixed bug in positioning of RadioGroup when scrollbars used
- added System,Log command
- rebuild Compatible to 080 now by default. Can be changed by command System, Comp80, {OFF|ON}
- changed Log icons are no more downloaded. They are as resources in WinBuilder.exe and are extracted on first use.
- fixed bug in FileCopy / DirCopy command with SHOW option
- fixed bug in Visible command
- fixed made z-order of components compatible to version 80

#### uploaded as WinBuilder\_110509.zip

- fixed made z-order of components compatible to version 80 (finally ?)
- fixed Font size and bold issue with themes different from classic
- fixed "Cancel' in "Create new attachement folder"

#### uploaded as WinBuilder\_110513.zip

• fixed - z-order fix sometimes broight interface display to hang

uploaded as WinBuilder\_110520.zip

• fixed - access violation in TextFile interface component

uploaded as WinBuilder\_110601.zip

removed [http://reboot.pro/14648/page\_\_view\_\_findpost\_\_p\_\_130037
 unnecessary SetEsc command]

uploaded as WinBuilder\_110602.zip

- fixed [http://reboot.pro/14477/page\_\_view\_\_findpost\_\_p\_\_129661 link function did not display first level linked scripts in project tree]
- fixed link finction displays non-script files
- fixed sometimes in the interface editor changes on grouped elements are not completelly saved

uploaded as WinBuilder\_110607.zip

• fixed - [http://reboot.pro/14477/page\_\_view\_\_findpost\_\_p\_\_130523 now different link behaviours with <linked path> vs <linked path>\.]

uploaded as WinBuilder\_110610.zip

- fixed [http://reboot.pro/14477/page\_\_view\_\_findpost\_\_p\_\_130616 bug in WinBuilder command line start]
- added NOWARN option to Exit command

uploaded as WinBuilder\_110612.zip

# [080] update (2010-03-03)

- fixed lost "search next" in source editor
- fixed nested macros now work

# [080] update (2010-02-23)

• fixed - hidden components in interface editor

# [080] update (2010-02-20)

• fixed - bug with missing last zero pair in REG\_MULTI\_SZ strings

### [080] (2010-01-15)

- fixed Bug ID #238: new scripts added to server not seen on download list
- fixed bug: command line 'If, ... ,If, ... ,Begin' causes double indent in log.html
- fixed bug: Some RegWrite errors have been logged as successfully processed
- changed RegWrite 0x1 tolerates syntax error (comma in string rather than escape)
- added If,EXISTMACRO command
- added Error when StrFormat calculates with uninitialized variables (gave accesss violation right now)
- changed Make GUI in Download Center simpler mit respect to 'Dummy User'
- added detection of Win7 to %HostOS%
- changed Set command: replace comma and space by escapes
- added AddInterface can optionally define an alternate interface
- fixed bug in RegRead multi\_sz
- added StrFormat,CTRIM and StrFormat,SPLIT now can have escapes as 'single character'
- added Variable %WBExe%
- fixed bug in DirDelete,FAST left temp files in root
- fixed bug in download center: On first show tree was allways built, regardless of 'green arrow' definition
- added RegAddMulti command
- changed improved syntax check with warnings / errors in RegAddMulti command
- $\bullet \ \ added \ \ RegAddMulti, HKLM, Section, Key, INDEX, SubVal, \%Var\%$
- added If,EXISTREGMULTI,HKLM,Section,Key,SubVal,Command
- added %RegDataType% variable set on every successful RegRead operation
- added RegWriteBin command to write wide strings
- added RegReadBin command to read wide strings
- fixed bug in scriptline parameter count check
- changed improved syntax check with warnings / errors in RegWriteBin / RegReadBin command
- changed Desktop path now read from registry

- fixed untranslated #\$x in script [Main] description
- fixed smaller bug with source editor autosave
- added error in RagHiveLoad when the hive is already mounted
- changed copy commands accept 'null' arguments instead of COPY, PRESERVE, SHOW, NOREC
- changed renamed RegAddMult command to RegMulti command (it handles also non-adding tasks)
- fixed show Options and Proxy tabs in initial download in Download Center
- changed Interface Bevel min size to 4 x 4
- fixed disabled 'Add new web server' in very fresh WinBuilder installation
- added RegWriteBinEx command to unchecked write any type from binary data
- fixed Bug ID #245: Dircopy different behavoir under xp and vista/7
- fixed Bug ID #246: Dirdelete,FAST
- fixed Bug ID #250: endless lop defining variables
- fixed In ShellExecute, Explore distinguish between file and directory.
- fixed Bug in RegWrite empty string
- fixed Bug in RegWrite command introduced by Copy Command 'Null' argument change
- added Set command also stores macros
- revised handling of empty values in RegWrite
- revised some more changes in handling of empty values in RegWrite

# [078 SP6](2010-01-03)

- fixed bug: Access violation on setting interface variables
- fixed bug in log.html generating. Loop,BREAK can bring "Index out of bounds" and lets WinBuilder hang
- fixed IniRead now translates comma in value to escape (troubles with TXTAddLine)

Release as 078 SP6 on 2010-JAN-09

## [078 SP5](2009-11-23)

- fixed bug: Unresolved \*.Link brought crash with 'invalid filename'
- fixed bug: Dependencies and Excludes did not work on \*.Link files
- fixed bug: No section shown in log, when started by button
- fixed removed senceless "Download disabled" message when file exists on "WebGetIfNotExist" command
- fixed bug in syntax check of "txtAddLine"
- added optionally run WinBuilder only in CPU / core #1
- fixed bug: changed ISOFile and TargetDir are not written to script.project

# [078 SP4] (2009-11-02)

- fixed bug: double comma when generating updates.ini
- fixed bug in download when path contained 'Updates'
- fixed bug in interpreting download levels
- fixed bug: no download tree on a fresh installation

Release as 078 SP4 on 2009-NOV-08

### [078 SP3] (2009-09-25)

- added AddInterface command
- fixed Bug ID #214: Deselection by keyboard produces error
- changed New Component Dialog now in center of WB window, rather than in center of desktop
- fixed Deselecting a complete folder containing a mandatory script lets build crash
- fixed Bug ID #222: wb078 sp2 Depend not working
- added Optional log at WinBuilder exit
- changed In download center, there is an option whether selecting an additional server rebuilds the complete tree.
- fixed bug in RegRead / RegWrite multi\_sz
- fixed bug in If,Online when running from command line
- fixed Bug ID #209: User-agent Windows Update: EmbeddedWB bug, using newer package
- added RegWrite 0x3 accepts one %Variable% as input value

Release as 078 SP3 on 2009-NOV-01

## [078 SP2] (2009-09-06)

- changed sepatate TNodeData unit with 'read on demand'
- added Download center now can select by Version, MD5, Certification
- fixed RegWrite multiple strings with comma in argument
- fixed Bug ID #217: Set,%var%,NIL,PERMANENT
- fixed Smaller issue with resolving nested variables
- fixed Bug ID #216: Access Violation when trying to trim empty string
- reanimated CheckListBox
- added change event to ComboBox
- added change event to RadioGroup
- fixed terrible bug with strFormat cutting long lines. Changed all according AnsMidStr Delphi commands to length of MaxInt

Release as 078 SP2 on 2009-SEP-24

## [078 SP1](2009-09-02)

- fixed source indenting disturbed certification
- fixed CopyOrExpand of single \*.ca\_ file does not work
- fixed leading and trailing spaces in variables were lost.
- fixed download issues when proxy does not allow DNS
- fixed Bug ID #211: Access violation
- changed certification issuer shown in log, when applicable
- added in Tools >> Upload tab, write certification string into updates.ini

Release as 078 SP1 on 2009-SEP-06

### [077 RC 3](2009-07-14)

- added some more syntax checks. Now e.g. comma in command parameter gives warning
- added When WB is not running as active window, ShellExecute,Open windows are not shown
- changed when there are troubles in downloading WB asks whether to continue or not
- changed removed PERMANENT option from AddVariables
- fixed bug in 'interface editor area' with non-standard WB size
- fixed Bug ID #205: StrFormat,Inc don't increment Letter
- fixed bug in concat of split lines
- changed warning mechanism on overwrite and parameter count: all is OFF by default
- added OnProcessEntry and OnProcessExit functionality
- added OUT capability for Run command parameters
- added StrFormat,LEN command
- fixed download issues with proxy
- fixed bug in OUT capability
- changed a lot of internet functions for work behind proxy
- fixed bug in OUT capability due to existing OUT variables
- added certification functionality
- fixed misfeature in IniWriteTextLine,APPEEND (empty line in section)
- fixed misfeature in IniAddSection (missing empty line)

### [077 RC 2] (2009-06-10)

- fixed In RegWrite REG\_MULTI\_SZ spaces are treated as separators
- changed In RegWrite REG\_MULTI\_SZ write empty key as hex(00 00 00 00) rather then not to create the empty key
- fixed Bug ID #196: Suggestions for improving UI (item #2 and #3)
- fixed Bug ID #197: RegGetNext access violation
- changed consolidation of all file transfer commands.
- added RegWrite warns on overwritintg different value
- fixed Bug ID #199: Run Failed to find section [AddShortcut] in file (bug in AddVariables)
- fixed Bug in Exec command which deletes GLOBAL variables
- added APPEND option to IniWriteTextLine
- changed Filecopy overwrite warning only with different MD5
- added Request Bug ID #201: Open Script Dir
- added Request Bug ID #200: Interface Editor: Component Selector
- added If,EQUALX command for case sensitive compare
- fixed Disappearing scrollbars in Editor / Attachments
- added Warnings and Errors in log.html header contain script titles
- added RegWrite 0x2 parameter accepted as hex number list
- added Option to warn before overwrite when extracting attachments manually
- added Extract of single file fom a CAB archive

published as WinBuilder 077 RC 2 on 2009-JUL-14

## [077 RC 1] (2009-06-03)

- fixed loop with 'Choose source' in case of only invalid source list entries
- added StrFormat,POSX and StrFormat,REPLACEX commands for case sensitive processing
- added ToolTips to GUI
- changed System, RESCANSCRIPTS does not longer restart WinBuilder
- fixed beta 3 introduced bug in FileCopy with wildcards

published as WinBuilder 077 RC 1 on 2009-JUN-09

# [077 beta 3 hotfix] (2009-04-26)

- fixed conflict between picture and 'show progress' in GUI button
- fixed 'List index out of bounds' in new Winbuilder installation with WinBuilder.exe only

### [077 beta 3](2009-04-26)

- fixed anchored "rename current file" in download center to window buttom
- fixed Bug ID #182: v.44 / WB077 beta2
- fixed Bug ID #188: CopyOrExpand warnings
- fixed (Some remaining issues) Fresh created interface components sometimes went lost.
- fixed ExtractAndRun with parameters
- fixed CopyOrExpand: M\$ Expand fature when files are compressed under a different name than expanded
- changed Log.html is now generated with W3C 4.01 compliancy
- fixed Bug in RegWrite with comma in value
- fixed Bug ID #192: \_Calculate, AutoIT Command
- fixed Bug ID #190: System, REFRESHINTERFACE
- added BartPE convert: put syntax error original lines into log
- added BartPE convert: Optional suppress Nu2Menu shortcut generation, when XPEINIT shortcut is present
- fixed BartPE convert: Some smaller bugs with progress and log.html
- changed Smoothed FileCopy command, added NOREC and PRESERVE parameters
- changed Interface components can be set like variables
- added Visible command to dynamically hide / show interface components
- changed Smoothed ScrollBar handling in interface editor
- added In single scripts, processed sections can optionally use ProgressWindow or not use.
- added Alternate interface section definable in [Main]
- fixed Bug in RegRead of binaries: First byte was duplicated
- added 'SHOW' as last parameter to ShellExecute: progress bar animated
- fixed Bug in RegRead of REG\_MULTI\_SZ with comma in value
- added StrFormat,SHORTPATH and StrFormat,LONGPATH commands

### [077 beta 2](2009-04-01)

- added Build progress shown in taskbar icon
- fixed Bug ID #177: Wait command bug 077 Beta1
- added Some visible user information in taskbar icon when WinBuilder builds in background
- fixed Bug ID #176: Script Count incorrect
- added Check commands for correct number of arguments
- fixed Bug in If command when comparing floating point values
- fixed Bug in proxy settings with disappearing 'z'
- added If,FileExist command now can check HTTP URLs
- changed Rewrote 'If,ONLINE' and 'If,PING' internal handling to Indifunctions
- changed Dynamical binding of psapi.dll allows start of WinBuilder under w2k
- changed Now WinBuilder internet functionality 100% independent from IE settings
- changed Download center optionally renames current file rather than overwriting it
- added System, SPLITPARAMETERS command
- fixed Bug ID #180: MULT/DIV replaced 32 bit integer by 64 bit integer
- added Interface editor has scroll bars by default.
- fixed Fresh created interface components sometimes went lost.

published as WinBuilder 077 Beta 1 on 2009-APR-26

### [077 beta 1] (2009-02-05)

- fixed dummy proxy entry in WinBuilder.ini, even there is no proxy used
- fixed ExtractAndRun did not resolve variables completelly
- fixed Bug in GUI FileBox component, setting %BaseDir% to last shown dir
- fixed Bug ID #157 'StrFormat,POS'
- fixed Bug ID #158 'StrFormat, SPLIT'
- fixed Bug in DirCopy with trailing backslash
- fixed wrong message text in IniMerge (error instead of success)
- changed System, RescanScripts in standard build does not restart WinBuilder
- fixed error 'File does not exist' which could occur on the very first start of WinBuilder in an empty directory
- fixed some unresolved variable values in IniWrite and MakeDir
- added Output Variables into log.html. Can be switched under Tools > Log Options
- fixed Bug letting download center crash on empty version number
- fixed Give the Exec command knowledge about API
- fixed Bug ID #165 'Processing scripts interface'
- fixed bug ID #164 'variable handling bug'
- fixed Bug ID #163 'Download progress bar'
- fixed Bug ID #159 'Tree view splitter bug reborn'
- fixed StrFormat,CEIL and FLOOR decimal point troubles, logically CEIL and FLOOR exchanged
- added Optional output of comment lines in log. Can be switched under Tools > Log Options
- added optional warning in log on unresolved (misspelled) commands. Can be switched under Tools > Log Options
- fixed Bug ID #167 Scrollbar markings in Interface editor have the wrong size.
- fixed Bug ID #169: WB GUI slightly to big for wondow
- fixed Bug ID #170: Options GUI to big for window yet no scrollbars
- fixed Bug ID #171: disappearing scrollbars in the treeview
- added Syntax highlighter for SynEdit with WB language
- changed Made Syntax highlighter customizable
- added Search / Replace functionality to source editor

- added Include the search / replace dialogs into 'foreign language' translation
- added parameters (#?) and escapes (#\$?) to highlighter
- fixed Bug in log.html generation with 'Begin' and 'End' in comment lines
- fixed bug with escape of escape (##\$p) in Set command value
- fixed Bug ID #174: Retrieve, MD5 bug
- fixed Source CDs of different projects got mixed
- added StrFormat, UCASE and StrFormat, LCase commands
- fixed Reanimated ability to hide interface component
- added script version to progress window
- added System, SaveLog command
- fixed Bug ID #175: OnBuildExit | OnScriptExit,BEEP
- added Interface component visibility controlled by CheckBox in iterface editor
- changed Set command now updates interface values immediatelly.
   Optional the additional PERMANENT argument to write into script
- added Interface scrollbox values now can be added
- reorganized Internal handling of Commands / subcommands from If ... To case ...
- added Syntax Highlighter gets word lists from WinBuilder source codeCommandLists.pas

published as WinBuilder 077 Beta 1 on 2009-APR-01

# [076] hotfix (2009-02-02)

- fixed Maximize bug
- fixed Proxy bug
- Rebuild Syntax highlight in editor restored
- fixed Bug (access violation) in Create Script
- fixed Bug in System, Rescan Scripts
- fixed Bug with Enable\_RefreshButton in WinBuilder.ini
- fixed Bug with Autosave in WinBuilder.ini

# [076] (2009-02-02)

unknown

### [075] beta 7 (2009-01-14)

- changed Dialogbox with timeout now has progressbar and seconds countdown
- fixed filebox in directory mode showed backslash at single drive letter (D:\) but no backslash at path (D:\mypath\subpath)
- fixed filebox in directory mode now shows last directory
- fixed indenting begin / end blocks confused on comment lines
- added enhanced WebGet functionality
- fixed 'No Network Support' functionality
- added GLOBAL and PERMANENT options to AddVariables command
- fixed Editor changes have been lost when switching tabs in source edit mode
- changed log by default in English, optional in user language (checkbox on Tools > Language)
- added API functions to magic wand
- fixed bug in WebGet with ftp protocol
- changed on IniWrite inifile is created if not exists
- changed Check button in source editor for begin / end indenting
- fixed script.project.sav files has been also treated as project root
- changed autosave logic: now with On, Off, On and Ask, Off and Ask (combobox on Tools > Options)
- fixed bug in indenting with 'Else, Begin'
- changed WebGet function enhanced
- changed options as first tab
- removed lblAdvLaunch to make room for additional option
- removed language entry !#1621=Launch a separate process for running projects.
- removed exit cross at the license tab to make tab look simpler
- added checkbox to enable visibility of Code Box tab (disabled by default)
- added language entry !#1625=Enable Code Box
- changed cosmetic improvements on Tools, now it's possible properly use all tools when window is maximized
- changed proxy tab moved to download center
- changed server tab in download center moved to frontpage
- removed buttons on server list (edit and visit webserver)
- fixed bug in evaluating RadioGroup

- added during build you cannot exit WB using the upper left 'x'. Build must be stopped before.
- added treeview to progress window, shown during project build (experimental)
- added System, GetFreeSpace command
- fixed escape of escape handling in TXTfile commands
- changed some smaller adjustements in User Interface
- fixed escape of escape handling in IniWriteTextLine command
- added some visual changes to progress window
- changed for mandatory scripts replaced red exclamation icon by locker icon
- fixed bug in Source window which let the \$sourcedir% variable sometimes undefined
- fixed bug in 'launch external processs'
- fixed bug in copying directories recursive
- added in Convert BartPE, added BartPE [strings] section to WinBuilde [Variables] section
- fixed I/O error 6 when exiting WinBuilder with folder selected (after editing a script)
- added warning when WinBuilder is started w/o admin privileges
- fixed bug that comment and API lines are not indented
- added Fast DirDelete
- rebuild complete 'unattended' design

### [075] beta 6 (2008-09-18)

- fixed bug in If,EXISTREGSECTION
- fixed bug in StrFormat, CTRIM if the SingleChar was an escape like #\$q
- added escape of escape. e.g. ##\$p writes the three characters #\$p rather than a percentage
- fixed bug in nested begin end
- fixed space in parameters
- fixed Bug ID #19: CPU 100% When double clicking an attachement
- added optional output of original lines in log
- speed in log html build, replaced if else sequence by case statement
- fixed RegRead no longer creates non existing key
- fixed bug in System, GETFREEDRIVE if variable was already defined
- fixed bug in nested begin end: Run and Exec disturbed the stack
- added System, ONSCRIPTEXIT, command
- added Custom logo size encrease, bug #89
- added STOP, ERROR, DONE, COMMAND as #1 to ONSCRIPTEXIT runs
- added System, ONBUILDEXIT, command
- fixed log listview now allows horizontal scolling
- added an log entry in the log listview can be copied to clipboard by pressing Ctrl+C or doubleclicking the entry
- changed internally split some objects from TScript
- added add some system exceptions to the ONBUILDEXIT functionality. shown as #1 = EXCEPTION tracked exceptions will be completed more and more
- fixed bug in ONBUILDEXIT when executing from a script button
- added codebox now with context menu 'magic wand'
- fixed bug in ONBUILDEXIT: Run in a different script was not executed
- added now the exec command also can have parameters like Run command
- fixed bug #92: dropdown list not saved
- added optional timeout for Message and If,Question
- fixed bug #94: edit of folder.project
- fixed bug #98: interface refresh issue when using XP style
- fixed bug #96: Now [Main] Description key may contain line breaks
- fixed bug #100: Save attachment directory

- added optionally hide 'Source' tab
- fixed bug #103: Running functions from interface editor
- added some new functionality to the 'Set' command
- fixed several bugs and misfeatures in GUI, especially when using XP-Style
- fixed bug in macro translation
- fixed bug #108: %ScriptTitel% not printed by magic wand
- fixed bug #95: [process] in script.project
- changed new project tree handling
- fixed several bugs introduced in beta 5 m
- added bug #107: language dependent interface
- added StrFormat,REPLACE and StrFormat,SPLIT
- fixed bug in If, EXISTREGKEY and If, EXISTREGSECTION: Non existing keys were created
- fixed tree navigation by key stroke
- added %WBLanguage% variable
- fixed bug in IniWrite 0x7: #\$S instead of spaces
- fixed bug in language dependent interface which created new interface section
- fixed smaller bug in BartPE inf processing: WB sometimes crashed
- fixed RegDeleteKey of a single value
- fixed bug #113: key navigation in download pane
- added optional confirmation of Build STOP button click
- fixed bug in If,Online
- fixed bug with #\$s in regwrite multiple strings
- fixed bug in interface editor when resizing a FileBox
- fixed bug #116: entering text by cut&paste in interface editor destroys the interface if there is a CRLF
- fixed bug #117: Return needed to add text by cut&paste
- fixed downloads are written directly to disk (first with a .tmp added) if the download succeeds it will be renamed to the final name, otherwise the tmp file is deleted
- changed escape handling in command dispatcher, hopefully providing something like bug #119
- fixed bug with space in macro definition
- changed wait logic of ShellExecute
- added ShellExecuteDelete command
- fixed bug in System, RefreshInterface
- added Verbose realtime WinBuilder file-logging mechanism implemented

(to enable it add "DebugLogEnabled=true" to WinBuilder.ini)

- fixed bug in interface: checkbox text unchangable or lost
- added StrFormat, CharToOem and StrFormat, OemToChar commands
- added Vista UAC warning to manifest
- added Proxy support, final version
- fixed some interface bugs introduced by internal restructuring started with beta 5 x
- fixed bug 128, Set command
- changed reanimated If,Ping command
- added WB size is saved in INI, and restored next start
- fixed some old WB 074 scripts in WB 075 appear with scrollbar
- changed logic of 'Size Borders' in GUI editor
- changed logic of 'Size Borders' in GUI editor: now measure scrollbar width
- fixed wrong color of GUI background
- changed some more internal structure, Interface editor
- added '\*.inf' can contain WinBuilder sections. [Process] is processed, [Interface] is used for building GUI, [Variables] are evaluated
- changed Source tab implementation
- changed Script editor form separated from WinBuilder's main form reimplemented large parts
- changed default language now available as independent in-exe-resource (editable via resource editor after UPX decompressing WinBuilder.exe)
- changed script interface gui can now be aligned exactly with the beveled banner above (use "pBevel1,1,12,-1,0,557,318")
- recreated escape of escape. e.g. ##\$p writes the three characters #\$p rather than a percentage (lost by introducing the VariablesManager class)
- recreated lost colored background in interface editor, when using nonclassic style
- fixed Fixed second editor (access violation, missing language, etc.)
- added 'escape of escape' functionality to 'Set,...,PERMANENT' command
- added Variables %ProcessorType% and %Wow64% to let scripts handle 64 bit specials
- recreated Magic Wand works in CodeBox
- fixed Variables %ProcessorType% and %Wow64% to let scripts handle 64 bit specials
- added Variable %Wow64Dir% to let scripts handle 64 bit specials
- changed truncate debug log file (WinBuilder.log) on start of WinBuilder
- fixed bug in Convert BartPE, introduced by inserting apostrophes to the

title

- changed Click on picture in GUI now may be some more Wow64 friendly
- added StrFormat,POS
- added New variable %HostOS%
- fixed Label font size bug with resolutions different from 96 DPI
- fixed bug in System, GETFREEDRIVE: a 'not connected network drive' was assumed to be free
- added System, REGREDIRECT
- added System, FILEREDIRECT
- fixed reanimated fixed folder view
- rebuilt old ShellExecute functionality
- fixed Access violation when there is no project
- fixed bug with lost file level on GUI edit
- fixed bug with labeled edit box on GUI edit
- added BartPE inf is skipped when [PEBuilder] contains WinBuilder=0
- fixed new bug with nested If commands
- added Indents in Begin ... End blocks
- recreated ShowSourceTab functionality which was lost in server transfer
- fixed bug with appearing 'Description' tab on script errors
- added small grid facilities in interface editor
- changed Wait command with process messages
- added If,RUNS command
- recreated lost language dependent interface
- fixed some new issues with refresh interface on xp-style
- fixed autosave of script source on program exit
- fixed bug in WebGet: If a file exists and WebGet of the same file failed with MD5 error, the existing file was deleted
- added On\_Exit and On\_Enter events to scripts
- fixed lost redirection in http download
- added System, GetFileSystem command
- fixed bug in magic wand: did not work in codebox after script edit has been used
- fixed correct WB startup coordinates when winbuilder.ini points outside desktop
- fixed Several lost download functionality
- fixed bug in magic wand: still did not work in codebox after source editor has been used
- fixed some new interface issues when working under xp-style
- fixed script was truncated when in attachement modus winbuilder has

### been exited

- fixed attachement 'delete folder' did not delete files, too
- fixed access violation after use of file box
- added System,IsAdmin command
- added System, GetEnv command
- reanimated If,License command

### [075] beta 5 (2008-07-07)

- fixed double 'beta' in form caption
- fixed unsaved change of download server constellation
- fixed default.txt is missing
- fixed 'Index out of bounds' when last download server is deleted
- fixed 'feature' how to remove custom program logo
- fixed some smaller issues with download servers
- fixed ExtractAndRun bug
- fixed internal variables like %scriptDir% can contain commas
- added 'Stop Build on Error' option
- fixed senceless 'v.' in progress window
- fixed sort order bug with mapped folders
- fixed ShellExecuteEx bug when opening documents by associated app
- added Option to name log.html with additional time stamp
- added Option to build log.html immediatelly after build w/o user interaction
- fixed Bug with If,(NOT)EXISTFILE/DIR checking empty floppy drive
- added Loop statement can run between characters, e.g. drive letters
- added Strformat,INC and Strformat,DEC also work on characters
- modified nicer output in log.html
- fixed bug with file type associations
- fixed Convert tool wrong popup menu
- added Convert tool remove file from list
- added suggestion how to have certain scripts be selected mandatory
- added RegDelete of values
- added 'Not' option to If command
- added StrFormat can extract path, filename and extentions from URLs
- fixed bug in 'Exit on Error' when extracting files
- fixed Minor code changes and corrections to clean the hint & warnings reported on compile time
- fixed bug 'system error occurs' in clicking log treeview
- fixed newly added web servers for download are not tracked in WinBuilder.ini
- fixed bug when running script from source editor
- fixed delete temporary files after extracting an attachement
- added optional automatic log after clicking the STOP button

- fixed wrong parent's node check marks when selecting with space bar
- changed RegDelete syntax
- changed starting to clean up language support
- changed remove 'charset' from log.html meta content tag
- fixed finally(?) wrong parent's node check marks when selecting with space bar
- some consolidation with respect to next beta release
- removed 'STOP' in interface editor when using themes other than XDPclassic
- removed 'protected' functionality
- added about 30 items to language translation
- removed new RegDelete syntax, temporarily use old syntax (see magic wand)
- changed new RegDelete syntax again
- fixed some smaller GUI adjustements
- added some more items to language translation
- added System, ERROROFF can get number of lines as optional argument
- added files to be converted can be Drag&Drop-ed from explorer
- added files to be attached can be Drag&Drop-ed from explorer
- added If,ExistRegSection and If,ExistRegKey
- added proxy support
- added 'mandatory scripts' functionality (replaces removed Protect functionality)
- cleaned internal If command logic
- changed WinBuilder always runs in CPU #0 only, also on multiprocessor and multicore systems
- added internal variables %ProjectTitle% and %ScriptTitle%
- added basic raw registry editor support
- fixed script was not shown if there was a folder with a name equal to the script's title
- fixed enable RegRead of expandable strings 0x2
- fixed decode macro names only as first argument
- fixed store changed text in a filebox
- added Show Progress Window also when processing codebox ot buttons.
- added several smaller adjustements
- added autosave option for source code editor
- fixed Changing source folder to a folder already on list
- changed parameter stack added for 'Run' and 'Loop' statements
- fixed bug occurring when a line contained a non-variable '%'

- fixed bug in If,QUESTION with space inside question
- fixed 'Use Path Rule' now unchecked by default
- changed some smaller corrections
- changed now nested 'begin' 'end' is possible. 'begin' stack added
- changed no flushing window on missing source
- changed dont show scripts of 0 bytes size
- fixed bug in System, GETFREEDRIVE
- fixed bug in previous / next selected: Changes were lost

### [075] beta 4 (2008-06-10)

- modified log button will always be displayed when using codebox
- fixed Else, Set, %var%, xxx bug
- added output of localized strings
- added StrFormat, Ceil and StrFormat, Floor
- fixed Some (WB 75 new) issues fixed with spaces in the %BaseDir% name
- added folder linking (link to script folder trees of a different project)
- added Retrieve, File Version command
- fixed bug in FileByteExtract (popup error if sequence not found)
- fixed project tree bug with deselect on collapsing
- added WBZip by booty#1 as standard de/encoding
- added System, GetFreeDrive command
- fixed RegImport bug
- fixed interface bug with unshown labels
- added System, Error Off command
- added Webget of FTP:// URLs

# [075 beta 3] (2008-06-09)

- added begin/end for If and Else commands
- fixed attachement context menu
- fixed multiple anchors in log
- added stepping through selected scripts
- fixed WebGet bug with redirection

### [075 beta 2] (2008-05-13)

- added 'variable names can contain variables'
- added Multiple 'Else,If' possible
- added 'Loop' statement
- added %ExitCode% for ShellExecute
- added System, Cursor, Wait / Normal
- added Expand Echo, Message to optional Echo, Message[, Warn]
- added button to open Download Center from main window
- added updates.ini can will open a custom website if specific info is found ([info] -> Website=<a href="http://myexample.com">http://myexample.com</a>)
- fixed greyed scrollbars area
- fixed bug with disappearing or moving components
- fixed bug with overwritten component, if wrong order
- fixed bug in RegGetNext
- fixed bug in script macro resolving
- fixed CopyOrExpand now works in W2000 host
- fixed Main treeview has a stable behavior
- modified Some cosmetics in log.html generation
- modified GUI editor fixes
- modified Log tab is only visible whenever it is not empty
- modified wb License is now visible by default under the Tools menu
- removed If,License and If,NotLicense commands
- removed IE dependency to allow wb work on Windows 2000 (no proxy support based on IE settings)
- removed Web and Help tab were removed to keep GUI simpler and quickier to load
- removed All built-in FTP upload functions were removed since most developer prefer to use own FTP clients
- removed Backup Tool was removed since it was rarely used

### [075] (2007-12-18)

- added Version number next to script description
- Fix bug #32 (Retreive cmd)
- Fix bug with RegRead MULTI\_SZ
- Implementation of REG\_NONE
- Log generation accelerated; Fix of 'truncated log lines'
- Add Generation of API format lines to 'Convert from BartPE'
- Separate unit 'ConvertBart'
- ConvertBart now can handle most 'standard' app scripts including a start menu entry extracted from nu2menu???.xml
- New unit nu2menu
- Fix bug #7 with Access violation on double click
- Fix bug in StrFormat: Now last parameter can be %variable%
- Add string functions RTrim, LTrim, CTRim, Left, Mid, Right to StrFormat
- Fix bug in If,ExistSection and If,NotExistSection
- New enhanced variables handling
- Add If,ExistVar and If,NotExistVar
- Add Else command
- RegWrite,0x4 now accepts four binary hex bytes in addition to DWORD
- Unload open hive after clicking the stop button
- Add ProcessMessages in RunLite loop
- STOP button now works also during Exec command
- Fix bug in RegRead 0x3
- Accelerated creation of log HTMLs

### [073] (2007-10-14)

#### beta 1

- Modified UI to keep tabs well separated and enlarge the size of the work window.
- Web tab now defaults to <a href="http://boot-land.net/forums">http://boot-land.net/forums</a> instead of <a href="http://boot-land.net/forums">http://boot-land.net/forums</a> wb.php to give direct access to the forums
- fixed bug on web tab which kept the progress whell rotating even after the page concluded loading
- Added right-click menu to add commands on the New Script tool
- Added right-click copy/paste/cut function to codebox and New Script Tool
- Fixed the CheckList component on the interface section
- Added standalone editor mode for editing script use winbuilder /open="mydir\myWork.script"

#### beta 2

- Added NativeEx server on the download list
- Corrected error messages that ocurred when editing the script's description in standalone editor mode
- Added option on standalone editor to run buttons and checkboxes
- Running a single script in unattended is much simpler, new syntax: Winbuilder.exe /run="c:\test\apps\myApp.script"

This mode will automatically try to find it's respective project file and read all default values from there if none is specified on command line

- Unattended mode can run files with any .project filename (example: winbuilder.exe %basedir%\Projects\LiveXP\LiveXP.project)
- Added stop button when running in unattended mode
- Added confirmation box when deleting attached files.
- Added button to run script inside the source code editor
- Added auto-save option inside source code editor
- Server list is directly updated from the internet instead of using built-in server addresses.
- Added button on the download tab to refresh list of available servers on the web

#### [beta 3]

- Added support for reading the default webpage from each web server when selected
- Changed icon for download button inside the download center
- Reverted back to default internal project processing
- Web server list is now available as a tab to keep things tidy
- Added two new commands to the IF function question and notquestion. Example syntax: if,question,"Continue?",echo,"Hello world!"

### [beta 4]

- Removed message that appeared after saving a script in the source editor save button automatically detects when script is modified.
- fixed bug that erased the contents of the script whenever using a spinedit object on the interface
- fixed bug that prevented loading any .script file outside the projects folder
- fixed Bug which appeared when selecting an error on the log window after running a script "Cannot focus disabled window"
- fixed bug that prevented loading .script files in editor mode using command line parameters
- added IniDeleteSection support for unicode files
- added IniAddTextLine support for unicode files
- added button to edit listed web servers inside the web server list.
- added missing <If, Question, "My Question?", Command> option on the right click menu of the source code editor
- added option to launch external editor in source code editor
- Corrected FileCopy command with progress bar it was only outputing the name of the filename being copied after completing the copy
- Corrected After clicking on button inside UI big play button is disabled
- Corrected System, RefreshInterface is not working after pressing button on script UI
- Corrected edit with right-click button on explorer erased everything except interface
- Corrected Text box won't remember changes made on UI editor
- Corrected Play button inside source code editor was not saving latest changes
- Replaced old inimerge code was replaced with newer version (please test)
- Replaced logo icon for source code editor tab

Replaced - navigation icons on main window

### beta 5

- Disabled SystemRefreshInterface is not executed when called from script UI buttons
- Corrected FileCopy was not allowing to copy a file from one location to another using a different filename
- Corrected Replaced all "wich" typo keywords by "which"
- Corrected IniDeleteSection was not working with UNICODE files
- Solved bug when encoding files using beta 4 incompatibilities between unicode vs ansi
- added Main script treeview is locked when changing selected status to avoid flickers.
- added Switch to TxtAddLine using "unicode" as last parameter on the append line operations will write with unicode support

### [beta 6]

- fixed bug that appeared when running the script inside the source code editor window
- Modified Single script upload button was moved to the "Options" tab inside the script editor
- Enabled MD5 tool to calculate MD5 checksum was readded on the "Options" tab
- fixed bug that "forgot" adding last folder on the "Create New script" tab

### [071] (2007-05-14)

### beta 1

- Removed MD5 box on the options tab inside script editor (download center always checks md5 automatically)
- Added support for running projects in unnatended mode

### beta 2

• Added /runsingle= switch which allows to run a single script

- Scripts can be displayed without state button useful for scripts not meant to be run (use Selected=None) on [Main] section.
- If all scripts on a folder have no state seletect (Selected = None) then the respective folder also omits any state
- Removed right-click menu from main window (became obsolete)
- Removed MD5 calculate tool from Script Edit --> Options --> Calculate MD5 checksum
- Removed language entries: !#1519,!#1520,!#1521,!#1522 related to MD5 calculate tool inside Script Edit --> Options tab.
- Added option to use wb without Download Center, Upload and Web tab for use under restricted network conditions
- Tools tab was made a bit bigger hiding the tab title
  - Script Edit tab title was also hidden along with script treeview to give more room to editor
  - Corrected bug which allowed to erroneously select more than one script on main window
  - Corrected minimum allowed width and height for interface checkboxes
  - Moved Paths and Log tab to the scripts tab
  - Added a Help tab which will hold help contents about wb, it will use a html file from %basedir%\Projects\Tools\Help\index.html
  - Corrected the back colors for some objects under some XP themes.
  - Recoded the Create Script tab to become more functional

- Added CopyExpand command using setupapi.dll (thanks to Peter and Smiley)
- Modified ShellExecute and ShellExecuteEx to support work directories as the 4th parameter
- Modifed ShellExecute and ShellExecuteEx to work without adding parameters, ex: ShellExecute,Open,cmd.exe
- Removed "Enter" from the keyboard shortcut list (it was being trigered after exiting windows screensaver protection )
- Corrected a bug which occured when trying to echo a section with brackets

- Recoded script engine to improve performance
- Removed words !#2057 and !#2060
- Fixed Refresh button wasn't updating correctly the Paths Tab
- FileDelete is now based on Win32API for code efficiency
- ShellExecute was improved to be (much) less CPU intensive while waiting for launched program to finish
- Added an error handler on the script engine now it won't abort when invalid syntax is found.

### beta 5

- Removed property on text labels that wouldn't allow using & chars
- Corrected a bug on regwrite introduced on beta 4

### beta 6

- Removed an unneeded setting which always wrote a locked=true value on the [main] section
- Recoded interface, file search/filter and settings to load much faster.
- Bugfixed error message when selecting to show log after running a project.
- Replaced external download functions with internal IE download function
- Bugfixed open handles when scanning directories

- Run command can support up to 9 parameters
- Added a code box window to test new commands under a specific project

- Added the option to print the currently viewed page on the Help Manual
- Moved Create Script tab to be the first visible tool
- Removed the option to lock the interface become obsolete

### beta 9

- Changed tab titles to be larger on script editor
- Small bugfixes
- Added a better web server manager
- Added missing translation texts

### beta 10

- More bugfixes and small improvements
- Limited treeview on main window to avoid excessive width
- Added more informations when using webget
- Changed default logo image for scripts without logo
- Added new parameter on FileCopy using "show" as last parameter will display the advance of the file copy operation.
- Fixed ExtractAndRun wasn't working correctly.

### beta 11

- Syntax errors are added on log along with the full syntax used
- Some bugfixings of previously added features.
- Added quick navigation buttons to allow pressing forward and backward while browsing scripts
- Modified the unattended box for running projects also added /debug switch to help evaluate used parameters
- Main window will automatically remember and open the last selected project in winbuilder.ini
- Modified the Interface handling when scripts are executed it was buggy under some conditions

### beta 12

 Corrected webgetifnotexist - wasn't checking correctly if a file existed on disk or not

<ul> <li>Corrected log window - it wasn't opening the log in explorer after setting this option</li> </ul>

# [069] (release date unknown)

### beta 1

- Converted ANSI\_CHARSET to DEFAULT\_CHARSET on all text labels to complete language translation
- Added a Beta Logo on the main window to allow clicking and acess the respective discussion topic

### beta 2

- Program will display window on top of other applications when starting up
- Removed edit button from interface to make UI less cluttered
- Removed fixed variables related to XP/2003 based PE projects (%targetsys32%, %targetwin%, %pedrive%, %pewin%, %pesys32%, %peprograms%, %tag%)
- When WinBuilder.exe starts up for the first time it will display the Quick Start page

- Added a small box on the Paths tab to list available projects
- When wb starts with no available projects it will hide unneeded tabs like "Scripts, Paths, Log"
- Fixed error window wich appeared when user pressed the refresh button on the web tab without being online.
- Removed references to obsolete PE drives on the right click menu of the source code editor
- Added a small correction when people used the download center with wb on a maximized window and would not be able to control it after refreshing the scripts
- Added support for levels inside project files
- Improved overall speed when creating updates.ini and index.html
- Added support for title on web server
- Re-Added text labels below the quick start buttons (Play, Tools, Refresh)
- Moved WebServer box to the lower right corner
- Added support for download profiles (Minimum, Recommended, Complete)

- Added support for displaying multiple web servers (good for quick browsing changes)
- Image on description tab will display a tool tip saying "85x90 pixels" to help users know the correct image size to use as script logo
- Added internal welcome web page wich is displayed whenever wb is started with no projects and unable to contact wb's homepage.
- Added new language terms to be translated for newly added functions

### beta 3a

- Completed support for multiple servers when downloading
- Fixed bug wich didn't terminated the download process when user closed the application giving sucessive error boxes per each failed download
- Added keyboard navigation on the project treeview on the Paths tab

### beta 4

- Box to select download profile no longer requires re-downloading webserver lists to re-order new selection
- Removed the buttons that restore a value with predefined settings on the paths tab
- Added a small project description on the same box where projects are presented on the paths tab
- Removed the "Save interface" button from script interface re-added the edit button with a new icon.
- Added an "Options" tab on the download center
- Moved web server box into Options tab inside the download center
- Added proxy support for download center
- Replaced the edit box for inputing a source folder with a combo box now it is possible to select a source from a list.
- Added a new download level called "Beta" this allows for scripts to be uploaded and only selected automatically if user wishes to use beta scripts.
- Added a warning box when user first selects the Beta download level
- Added the option to upload scripts to <a href="http://beta.boot-land.net">http://beta.boot-land.net</a> whenever no webserver is available used as generic upload place for beta testings.

### beta 5

Download Level box will remember the last selected level

- Each web server will have the title of the web path if no title is available
- Support for Proxy SOCKS was added
- Modified word !#1486 from "Download Type" to "Profile"
- Removed words !#1216 (Projects), !#1217 (Target), !#1218 (ISO) to keep them as fixed values on all languages (They represent common folders)
- Added FindClose on each function which used FindFirst to avoid open handle issues noticed when using the backup tab and some file search routines.
- Fixed bug which prevented to download last placed webservers if there was not selected script on the first webserver

- Added mobileos.boot-land.net
- Readded refreshing code to make interface display correctly objects
- Fixed bug which made Add object window stay behind main form, also fixed the attachements add files wizard.
- Copy & Paste from source code editor will remember the last copy operation before adding a new command

- Fixed small bug which didn't added paths with spaces on source folders when using the Source wizard.
- Boxes to add description on folders are a bit bigger
- When writing folder.project files, a download\_level=0 is also added to ensure the file is downloaded by default on the download center

### [068] (2007-01-30)

- Renamed Download Center to Updates Center to add the upload service
- Modified the links menu on the options tab to include a link to VistaPE and Winimize
- Corrected the ExtractAndRun template from the right-click menu on the source editor
- Clicking on folders from treeview will display sub folders and files inside it
- Added an ftp client on the tool box
- Filecopy command will create a new directory if the target dir doesn't exist already

### beta 2

- Pressing enter when typing the web server adress on the options tab will act as pressing the save button
- Download method rewritten and improved now it won't seem to be hanged when moving the window around
- Removed the "Open" button since it wasn't very used now wb will look on inside the first folder on the projects folder for script.project and add them as projects automatically
- Section [projects] inside winbuilder.ini has become obsolete

- "Create new updates.ini" tab was renamed to "Upload Config", allowing to create updates.ini/index.html files and add needed values for uploading scripts using the file tranfer protocol
- A few more bugfixes on the download method, safer behavior.
- In the middle of the download process it is already possible to close the application, aborting safely the download.
- updates.ini and index.html are completely independent from fixed web
  paths using using relative web links meaning that it can be placed/moved
  to anywhere as long as the project/script found on subfolders are moved as
  well.
- updates.ini and index.html are now created and placed inside the projects folder - making it easier to just drap and drop everything on the new web location

- it is possible to preview how index.html will look when uploaded just open/doubleclick it from disk to see the result
- Added WinBuilder's License Agreement inside WinBuilder.exe and a button to view it on the Options tab no more extra files needed to distribute, wb can now be distributed as standalone exe.
- Backup tool create button will warn whenever the projects folder is not removed as specified by the option to remove this folder after creating a backup
- Removed the right-click option to remove projects from the main window
- When pressing the Play button without selecting a valid source the Source path edit box will flash a few times as a reminder of the need to input a value
- Added right-click option on the main window to upload scripts straight into the web server.

- Moved the web path box to the update center
- Added files remaining to download while downloading
- Source folder input box will "flash" a few times if source is empty and user presses "Play"

### beta 5

- Added support for multiple servers and predefined server list when no updates.ini is available
- Added option to skip the FTP upload test when typing new ftp locations
- Improved significantly the readbility of index.html using CSS Also reduced overall code size
- Added support for automatically creating new webservers along with upload
- Added new button on the interface to quickly upload script

### beta 6

• Small bugfix on the webservers available when no updates.ini was available

### beta 7

• File sorting is no longer dependable on NTFS, meaning that it is compatible

- with other compatible file systems like fat32
- Fixed a bug that prevented files (not scripts) to be downloaded
- Added the quantity of downloaded bytes while downloading files
- Fixed the progress bar that follows the download of the current file
- Added a small animated image on the download center
- Added "IniMerge" INI command to merge the differences from one inifile into another, syntax: IniMerge, "UpdatedIniFile", "InifileToUpdate"
- Main window position has been changed to default windows position

- Added a web browser based on IE engine
- Created a wb homepage on boot.land.net with contents to help new users
- Modified behavior to start the web tab instead of the download tab when no scripts are found
- Modified FileCopy/DirCopy to ensure that the target dir is created if not already existent
- Started implementing an external page for members to login and quickly view new topics/replies amongst other posting on boot-land.net good to make it easier to keep up to date with news.
- Added support for %projectDir% and %script%

### beta 9

- Added a "nudge" when user presses "Play" with an empty source dir.
- Added option to choose if log.html was opened by browser after being created
- Modified icon for Download Center
- Added wbChat a small tool to chat around with other wb users
- Modified Play, Tools and Refresh button from main window
- Added "Stop" button on web browser
- Modified FIleCopy to ensure that only a folder name is provided as target path

- Added support for Enter key inside web browser, theme style and automatic form filling.
- Added display of author name while downloading

- Added safety check wthen downloading files from the download center first it's downloaded to extension .web then it is renamed as original file if
  passes MD5 integrity check
- Improved the safety check when trying to upload a single script without valid configuration or not connected to the ftp account
- Added the individual file upload progress bar when uploading a new web server
- Removed the previously added wbChat from beta 9 wb's homepage chat is a good replacement for the moment
- Added "Stop" button for individual file uploads, also fixed the respective progress bar
- Added the option to remember last position of the main window this
  option will be ignore if no winbuilder.ini file is found (to keep the root
  folder clean)

- Added safety ping check to homepage adress to handle situations when wb is not online
- Added the same safety check to the download center
- Added a "Source Wizard" safety check and validation that ensures that specific files are found inside the source before continuing the project build.
- Moved the Options tab to the tools section
- Moved the Paths to the last tab position

### beta 12

- Tool section button can show/hide respective tools tab
- Added an icon next to the source description on the source wizard
- Modified progress tab added the name of the script author and a few small UI changes
- Modified error icon displayed on logs to a red X to make them easily identifiable
- Added UI language support an editor was also placed on the tools tab and will write text files to projects\tools\languages

### beta 13

• Added more support to allow internal program messages to also be

translated

- Fixed bug that forced lower case on variables like %projectdir% (should also fix the ftp folder create in lower cases)
- Modified the "simple" function to find files to support vista and ignore system and hidden files wich are protected.
- Added spaces on some labels to ensure that they are displayed correctly under vista
- Reverted the load hive procedures to use a shell wrapper around reg.exe to ensure that hives are loaded correctly under vista with administrative permissions
- Added vistape.boot-land.net/project to the list of available servers
- Fixed the ping command, the previous method was not working under some conditions
- Merged FTP client tab with the upload tab, creating a set of 3 tabs to allow using common definitions when uploading files
- Modified fonts to support other languages
- Increased the size of tab titles to make them more easier to navigate
- Fixed the blank tab on the web tab when first acessing the page (delay was cause by a ping to the host to see if it is available before navigating)

- Fixed bug on right-click source code menu selecting DirMove would place a DirCopy command
- Further improvements on the FTP section
- Some more UI language bugs fixed
- Modified FTP section the only place to set user settings is on the respective settings tab all FTP connections share the same settings (individual script upload, ftp client, new webserver tab)
- Fixed a bug wich prevented script.project files of displaying the correct version, author, contact and credits
- Credits box won't accept enter key to prevent unwanted end of line.
- Removed entry !#1274 from language list
- Fixed a language bug on setvar
- Fixed a bug when pressing the right-click button on the main window
- Fixed language tool tip bug on script editor instead of show "Remove logo" displayed "Visit this page";)
- Added a \ on the path when uploading single scripts
- Removed the right-click option to upload a script to avoid conflicts with the

- option available from the script interface.
- Fixed a bug wich kept the upload progress bar on the script interface visible when the upload was aborted by errors
- Users can now download files from the download center, use the web tab and then return to the download center withouth reseting the connection

- When a user saves ftp settings with an empty path the program will add at least the root folder as start location ( / )
- IniRead has been modified If an INI key is not found on a file then it will output an empty variable instead of an error message
- Fixed the function to upload and create web servers two errors debugged, first was the lack on correct details when logging in, the second was time-out too short (wasn't working on slow dial up modems)
- Bug fixed the extra \ that appeared on section name when uploading individual scripts, causing them to not be displayed as available on the download center

### [067] (2007-01-28)

- Fixed a bug on Exec command wich outputed wrong result messages.
- Added the hability to run section [process] inside the project file (script.project)
- Corrected the System, RefreshInterface message to: "System Refreshing interface after finishing processing"
- Wizard window for encoding files was not centered when using desktop with two displays - Fixed
- SetVars now support automatic update of all variables with new value
- Encoding tab presents filesizes in Kb and Mb (files inside scripts need to be re-encoded to reflect these changes)
- Links in the support tab have been updated
- Added quick-close, whenever the Escape key is pressed on the main window will quickly exit the program.
- Added a backup tool to make safe copies of Projects, Archive, Tools and ISO folders
- Added right-click menu options for expanding and colapsing sub-items on main window.
- Added a download tool for browsing files available on servers.
- Fixed a typo on webget wich made the secondary progress get invisible instead of showing the download progressing
- Corrected some GUI elements
- Download tool can now browse subfolders
- Changed designation of encoded files to attachments
- OVerall GUI changes in icons, logo, button position on Attachments tab.
- Added a button on description tab to visit website link
- Fixed the script and link filtering wich prevented the scripts to be mixed along with linked scripts on normal view
- Different icons added for either scripts or links.
- Regwrite now supports the creation of single keys with no data
- Removed the option to update winbuilder.exe because it was obsolete.
- Removed the option to update individual scripts until a new handling method is added.
- overall logical speed optimization in .script engine
- Cabexpand is using cabinet.dll to extract files instead of wrapper to expand.exe (faster)

- program's process priority is changed to HIGH when processing scripts, the same applies to child process launched by shellexecute.
- webget and webgetIfnotExist support MD5 checksum check to ensure 100% correct downloads
- Added command "regimport" to allow importing a .reg file straight into the local registry
- bugfixed the internal inf2script tool to correctly add "" on each converted value on registry keys of type 0x3 and 0x7
- DirMove has also been improved to allow moving files and folders silently
   allowing to use \*.\* to only move files and subdirectories from a folder
- Echo command was also bug fixed was not translating variables
- Clicking to disable or enable a folder will also reflect this value on all scripts inside
- Added a new tab named "Tools" where "Backup", "Create new script" and "Create updates.ini" tools where placed
- Added webupdate, allowing users to view and download the latest scripts and projects
- Added a new command to refresh the variable list as an option of the system command - available on right-click menu on source code
- Projects no longer accept empty %sourcedirs% displaying a message box for users to select a valid source folder
- Tools were moved to a sub-folder of archives to take advantage of webupdates, %tools% variable was also updated to reflect changes
- Replace the individual script update button with the option to run the selected script
- Added a global webserver edit option for using updates from alternative webservers
- Cabexpand was reverted back to the previous expand.exe wrapper until a more stable code can be used instead.
- bugfixed webdownload was not decoding filenames properly and wouldn't download filenames with spaces
- when winbuilder.exe is started for the first time without any project available will jump straight to the download center tab
- bugfixed regwrite wasn't supporting empty 0x1 entries wich created registry section keys
- bugfixed regwrite 0x3 keys were also not being correctly handled
- bugfixed open handles when looking for files no more locked directories
- Methods for searching, filtering and categorizing files were improved program starts significantly faster

- Disabled the old method of sorting scripts under the options menu to avoid conflicts with new method
- Refresh button will also reset the script window, whenever possible it will also try to re-select the previously selected script
- Edited the description on the Paths tab to warn first time users that each project holds independent values
- Re-added dependencies with the difference that now it is needed to add the path to the needed files or scripts please use %basedir% to avoid to fixed paths (check explorer.script as an example)
- Added the option to add scripts wich are incompatible when one script is selected (example: ramdisk vs fbwf), also valid to any sort of file if found.
- Created a new section in tools where it is possible to convert files into native .script language, removing the respective buttons inside the source editor. BartPE plugin conversion is still incompleted it's preferable to use external tool to retrieve accurate translations in the meanwhile.
- bugfix on filesearch search is not case sensitive anymore
- Removed the Archive folder and moved all static references to the Projects folder
- when clicking on folder, paths tab is now updated with directory values from respective project
- Fixed error wich caused "abstract error" when trying to use interface buttons
- Fixed error wich caused "Read error" when trying to use the "System,reScanScripts" command occured when pressing button "Restore Options" on myOptions script from NativeEx
- Derivated from the above fix, a command line parameter was added to allow select a script from startup, syntax: winbuilder.exe restore level gender "ProjectFilename" "ScriptFilename"
- .project files are treated as .script on download center
- Read feedback button was fixed
- Added progress bar to track the overall progress of the download process
- folders wich contain script.project files will use these files as description usefull for describing projects just by clicking on the respective folder
- when a script on local host has a bigger version than the one available to download from the web server it's not marked for update.
- Added support for using localhost even when not connected to the internet (or any network connection at all) - and use a download server from a local http server (localhost) - great to speed download testings, e.g.: http://localhost/winbuilder

- Added a button to stop the download process
- When running a single script script.project variables are also loaded
- Readded command "ShellExecuteEx" similar to ShellExecute, but will launch the program and continue to run the script
- Fixed a bug that prevented the last file available on updates.ini to be downloaded
- when typing a new download server bug occured under some circumstances
   the autosave was replaced by a button next to the text box to avoid any issues
- Fixed Open Handles that locked folders which occured creating a new updates.ini from the tools tab
- Added filesize on .script .project on updates.ini description
- Added a text label indicating the number of selected files and respective size in the download center
- Removed list of processed sections in log.html
- log.html was redesigned from scratch, using tables, colors and graphics to make it more readable
- Removed splasscscreen to make startup faster and reduce overall size of exe file
- when the user tries to start a project with an empty source dir a message box will ask for the user to input a source folder now it will also open the Paths tab
- Removed hard links from updates.ini now all script downloads are based on the web path to updates.ini, meaning that it is now possible to move all files including updates.ini to another folder and keep downloading it.
- Added some code to automatically generate an index.html file based on updates.ini usefull to also upload to the folder and allow an easier navigation, also created with updates.ini from the tools box
- Added the option to select negative levels for scripts this keeps them hidden from the main window

## [052] (2006-10-07)

### beta 1

- Corrected the text label "Stop" on the progress window
- Removed a TScrollbox component from the main window less flickering and faster interface GUI loading
- Corrected the image on the "Open" button on the main window image didn't look good on windows classic theme
- Added Logo Image support for JPG, GIF, ICO, WMF and EMF files. Animated and transparent GIF's can be used.
- New mode of browsing scripts has been added
- Script explorer is now working with autoexpand feature. (much easier to navigate through scripts)
- Support for bmp images

### beta 2

- Fixed the Script Reload button on the source code editor
- Removed auto-load script from disk whenever editing the script in the source code editor
- Changed default script (de)selected icon
- Added option to select script brownsing mode

### beta 3

- RegRead has been corrected now %vars% are working as supposed
- StrFormat now supports the options "path", "filename", "ext" and "hex" (check menu on source code edit window to see syntax examples)
- Progress window had a few cosmetic changes, new stop icon, frame titles in bold
- A new license mode is available inside scripts allow users to accept a license before using a determined software.

- Fixed interface bug wich prevented from saving interface options
- Interface drawing is done faster (bug when maximizing makes a few labels

- disapear temporarily)
- Added "Open with.." button on encoded folder tab
- Added option for registering filetypes on startup
- Removed line numbers from log.html

- Fixed the new script tool now converts %BaseDir% without issues
- Added quotes on echo statements derivated from the bartPE plugin convert tool - should avoid hangs
- Added safety checks when switching scripts (to avoid interface overwrites as noticed by smiley)

### beta 6

- title bar displays full version information
- text file component on interface supports rich text format (\*.rtf), allowing to use text colors and formatting
- images in interface are supporting more formats (\*.jpg;animated .gif;\*.bmp;\*.ico;\*.emf;\*.wmf)
- buttons support custom images (\*.bmp)
- Some redundant code removed, UPX'ed winbuilder.exe has reduced 140kb from 780Kb to 640Kb
- Main window can be resized without text labels on interface disapear
- Process window no longer appears when processing a section called from a interface button

- Scripts without [interface] section won't output an error
- Added If,NoLicense option to handle cases where the user license is not accepted
- Added "Halt" command to terminate the current process. syntax: "Halt,message"
- Process window resets all values after processing will give a "cleaner" look when processing again on startup
- Updated UPX to version 2.02
- Not updated AutoIt to v3.2.0.1 autoIt.exe weights +400Kb while previous version is 122Kb, added autoIt help file for license support

• FileCreateBlank will no longer overwrite an inexistent file - will output an error and continue processing

### beta 8

- Improved refresh repaint on interface labels won't disapear when selecting another tab (still needs some fixing)
- Filebox was only allowing to select images instead of wildcards (.)
- FileCreateBlank will behave as build 051 (erase existent file and then create a blank file)
- CAB expand command (wrapper to expand.exe) fully supports non-ascii characters
- IniWriteTextLine will write correctly lines on sections, always adding on the first line
- External script editor button has been disabled
- "Apply same path to all projects" option has also been disabled

### beta 9

- Added option to auto expand items on navigation window
- Fixed error when enabling .script files in advanced view mode
- Fixed a bug wich prevented projects from reading their descriptions

### beta 10

- Script checkboxes will wordwrap on their titles
- Charset has been changed to default wich allows the use of different codepages
- Checkboxes can process sections whenever they are clicked.

### beta 11

• when running (double-click) encoded files from WB the main application won't freeze as in previous versions.

# [051] (2006-09-22)

### beta 1

- Fixed the IF command option that checked if an Ini file section existed or not.
- Disabled popup after pressing the save script button
- Increased the editing window on the source code editor tab
- Added the option to use an external editor on the source code editor
- Added Call a command similar to RUN, executing a script using the script variables (including interface vars)
- Fixed bug in bart plugin convert tool comments and empty lines are now supported

### beta 2

- Stop button no longer freezes whenever shellexecute command halts, terminating the launched application
- beta "Call" command was renamed to "Exec"
- Added a menu for creating new script files.
- Added "Options" tab.
- Added option to disable splash screen
- Winbuilder will no longer crash whenever WinBuilder.ini is not found, and will use default values

### beta 3

• Added dependencies on script files (much improved on beta 4)

- Added dependencies on script files
- Added command "encode" wich can be used to encode files inside script or text files (supports wildcards)
- Added command "system" wich allows specific actions outside the script engine - like the hability to refresh the script interface or rescanning all scripts again
- FileCreateBlank will now create new directories and a new file if none exist

- (and if the path is not read-only)
- HTML Log as been improved to add more details in builds like version, %basedir% value and a few more system messages
- Corrected "View Last log" button
- Rearranged a few GUI elements on the progress tab
- Added checkbox to use the same source directory for all available projects
- Added hint for images in script interface saying "Click on image to view in full size"
- Added keyboard support for navigation on the script window. Space=disable/enable, Enter=Run project, Up/Down=change script
- Added keyboard support for navigation on the log window. Up/Down=browse log messages
- Form resizing was disabled to avoid GUI problems under 120dpi

# [050] (2006-08-21)

- Generated log.txt files now support a results filter to ease diagnostics
- Log files are generated in html format
- Splash screens and quotes were replace by single logo designed by TheHive
- When pressing the update button on a project it will search and update all scripts within
- Added multiply and division operators to StrFormat
- Section [Variables] inside winbuilder.ini will be applied as variables for all projects.
- Overall component drawing speed was improved

### [049] (2006-08-17)

- Fixed a few invalid entries on the right-click menu on the source editor
- Changed the default process level when file blank from 3 to 4
- Added MD5click to ensure the integrity of downloaded files
- Added net update support for .link files
- Added AddVariables a command that allows to load variables from a section in an ini file
- Added special character #\$p that will be decoded to % (usefull to use in batch commands)
- Section [Variables] on script.project files now holds global variables that will extend to each script
- Added a global routine to handle exception errors (still needs to be improved)
- Added a secondary progress bar on the progress window to follow the execution of sub scripts when using the Run command
- Removed some unneeded components making the application lighter
- Increased the size of the code edit window text font now it's on standard size
- Added parameters to the RUN command, they are available as #1, #2, #3, #4
- SetVars has been improved
- Improved the rigth click menu on the source code editor
- The currently edited script interface is saved whenever someone tries to run it.
- ScrollBox can be resized to smaller sizes

# **OpenBuilder Releases**

# [048] (2006-07-27)

- Fixed misplaced image when processing scripts in maximized mode
- Processing image is now rescaled and proportional to fit a smaller frame
- Fixed plugin support: the registry hives were not being loaded as supposed
- Exchanged the position of sources tab with options tab in the script windows
- Fixed ExtractAllFilesIfNotExist, it wasn't working.
- Interface web links are using Verdana as the default font
- processing sections from interface buttons will no longer push automatically the view for the log window
- Added %username% to reflect the current name of the user logged in
- Added %userprofile% where all files and settings from the current user are located
- Added %tag% on the right click menu on source editor
- Added %day% %month% %year% to support date handling
- New interface component: FileBox, wich allows to select a file
- Exit command will only force exit from current script and will continue to process other scripts
- Added %version% to display the version of openbuilder
- Added a new interface component: Filebox, wich allows to select files or folders from the interface.
- Fixed right-click menu options "Uncheck selected" and "Check selected" they weren't working properly
- Added new command StrFormat, with four functions: date to format a string into a date, and bytes to convert a value in bytes to easier to understand values like Kb, Mb, Gb.., inc and dec to respectively increase or decrease the value of a variable
- Fixed a problem when trying to change the process level on links
- OpenBuilder window no longer freezes whenever it's running scripts

### [047] (2006-07-18)

- Removed the wiki folder, since this feature is now hosted online ( http://www.boot-land.net/wiki)
- Fixed a bug that caused the application to crash when wiki folder was not found
- Created an Archive folder to store common scripts
- Added the .link file feature, wich allows to use text files with extension .link that point to the real script file - a good way to avoid double files, and all changes made on a link file will immediately be reflected on the linked file itself - except for the status of enable/disabled switch, that allowing to have several different profiles using the same script.

It will preferably use %basedir%\Archive folder to avoid static path problems.

- Images in description are no longer automatically streched or proportional causing images to be more vivid and look better
- Added an Archive project not meant to be used as project, but rather as a resource to edit available scripts inside it.
- Added an option to erase the target folder (placed in the makedirs script) as suggested by Olri
- Added the hive, explorer and ramdisk scripts sent by Ovi
- Added projectInfo script by psc
- Fixed the result output on IniDeleteSection
- Right-click option on retrieve command was not working for the file and dir options - Fixed
- Added the update service for scripts (server by default is <a href="http://www.boot-land.net/updates">http://www.boot-land.net/updates</a>)
- Added a new tab on the script windows, called "Options"
- Added an option to use custom update servers for specific scripts
- Added MD5 digital signature checks stronger algorithms will probably be implemented soon.
- Updates can choose to wether or not include MD5 protection (recommended to ensure reliable downloads)

# [046] (2006-07-15)

- Replaced old icon and logo by new ones designed by TheHive (thanks!!)
- Corrected all links to point back to the new forums location
- Corrected the location for updates on openbuilder.ini file

# [045] (2006-07-11)

- Added %tag% a random variable number from 0 to 99999
- Fixed a bug regarding the message command
- Plugin support replaced \" by " on filecopy commands
- Plugin support Added the missing quotes
- Copy and Expand script has been rewritten by psc using new method (available on develop folder)
- Explorer and Ramdisk script by Ovidiu are also available on the develop folder
- New command: retrieve, file, target, %var% opens a dialog box to select a file based on target
- New command: retrieve,dir,target,%var% opens a dialog box to select a dir based on target
- Load and Unload hive no longer requires reg.exe, openbuilder **should** be

able to run from win2000 and above, I hope it fixes DirkGently42 issue,but it's still missing to add native support for importing .reg files into registry - this might pose a problem. I intend to add full .reg support soon, but REG.exe is still used to provide this feature on the hives script.

 Replaced tool 7za.exe by 7z.exe (provides .cab support and is sized in 52Kb)

# [044] (2006-07-08)

- Added more lines to the starting quotes
- Added a 2 second delay before showing the main window
- Corrected the standard and nanoXP copy and expand scripts
- Corrected the hives script interface
- Added a license for openbuilder.exe
- Removed comctrl32 from the UPX script (thanks for noticing psc..)
- Fixed a bug when switching throught scripts with locked edit mode
- Reordered the scripts to keep distinct actions more separated
- Corrected the check for using a proper source path on each project.
- Added routines to handle mouclass.sys issue
- nanoXP can now support win2003 (ISO bigger but works) thanks again psc

### [043] (2006-07-04)

- Updated UPX to version 2.01
- Some minor GUI improvements
- Renamed scripts on the standard boot folder
- Added the nanoXP project
- Added 10 priority levels
- Added the wiki method
- Updated icons on the New script window
- Added some of the wiki content
- Fixed progress window (description in maximized mode)
- Fixed Source Edit mouse was always on edit selected mode
- Changed order in paths and log tab
- Press save button and return to source editor was fixed
- Added a message box when pressed the save button on the source editor
- Added licenses for most of the external software included in the distribution

### still missing to add some licenses.

- Fixed the description issue on multiple projects (it wasn't handled properly)
- Added an image related to the processed script when it's being executed
- Downgraded qemu back to 0.8.0 for compatibility reasons
- Hidden the new script wizard will be completely rewritten in future builds
- Remove the added line when inserting commands from the rigth click menu
- Hidden the %pedrive% menu until it is implemented properly
- Added FileByteExtract a command that will search for a byte signature on files and extract a byte portion to another files when found
- Readded the RAMBoot script to allow win2003 builds run in RAM
- Fixed the file list window on encoded file's window
- Added a message when trying to switch from edit mode to user interface without unlocking first.
- Added an english software reg file (to replace portuguese version)
- Updated extensions.rc file in xoblite to avoid the "check for updates" dialog box

### [042] (2006-06-26)

- Added the option to (de)select all files within a project to the right-click menu
- Some minor GUI fixes
- Added 7Za.exe to the tools folder
- Updated QEMU to version 0.8.1
- UPX'ed gemu dll's and gemu.exe to save disk space
- Removed txtsetup.sif from the hives script (not necessary anymore)
- Added a script description window on the progress tab
- Fixed the net update function
- Readded the auto-update function
- Rearranged the visual elements on the paths box
- Added the UPX script, wich will gain around 3Mb from compressing some system32 files
- Script engine is more efficient takes less time to process each command
- Added the lines related to Hal. on txtsetup.sif to prevent BSOD on VMWAre and Virtual PC emulators
- Added small text file for description on each folder to better explain their purpose
- Fixed the process level selection box on the script description
- $\bullet\;$  FileCopy no longer requires an "\" on the end of the second parameter (dir)
  - it can be ommited

# [041] (2006-06-23)

- Fixed some shortcuts on the source code editor wizard some %variables% were not correctly linked
- %PEdrive% variable was not being processed Fixed
- Rewrote some code on the standard scripts to use the [%BaseDir%\Tools] Folder
- Added the ISO path creation as sugested by psc
- Added a %Tools% variable that refers to a [%BaseDir%\Tools] folder

this is where most GNU tools will be placed in order to save space on multiple projects (avoid the need to double binaries)

- Added support for multiple extraction of files on the encoded tab still some more feature needing to be added in the future..
- Added UPX 2.00 and expand (from ReactOS) to the tools dir
- QEmu and mkisofs were also placed in the tools dir

# [040] (2006-06-22)

- Added basic support for bartPE plugins
- 0x0 entries are ignored, at this point it can safely be used on simple plugins
- Addline entries in plugins will be ignored. (next release will probably include them)
- Added a convert Plugin to script syntax button on the source editor
- Some GUI improvements
- DirMake is now forcing to create dirs and subdirs if they don't exist
- Changed the delay to the project switch in the navigation window
- FileCopy has changed: No longer is needed to add the target filename only the target dir is needed.

Old syntax: [FileCopy, c:\text.txt, %targetdir%\text.txt] is now: [FileCopy, c:\text.txt, %targetdir%\] This change was forced by the use of the win32 API native copy functions - to allow compatibility with plugins

- DirCopy is also using Win32API functions this allows a much more stable behavior
- Added %ISOfilename% it will output only the filename of the ISO file
- The source editor saves and loads the respective file each time it is used to avoid manually save or load from disk
- Added line count in the standard code edit window
- Reduced the font size on the source editor to better view the commands
- Fixed the window title wich kept repeating the build number
- Fixed the bug that eliminated from the main window the last item on the list
- Fixed the bug that kept writing the locked state of components on the file (only written when true)

# [039] (2006-06-18)

- Removed ShellExecuteEx it caused problems with the multithreading feature (yet to be added)
- Removed the option to delete scripts from the main window.
- Added the echo's back on the messages window they now output all process progress
- Added support for the AddLine and DelLine sections in bartPE plugins
- GUI has been improved on the script progress window

# [038] (2006-06-18)

- Fixed another bug on extractfile it wasn't supporting paths with spaces
- hidden the option to search for updates on start, it was a bit unstable as it would prevent the program from starting in case an error occured in the network connection

### [037] (2006-06-16)

- Fixed extractfile it wasn't working properly syntax is: ExtractFile,%scriptfile%,folder,filename,folderToExtract
- Added a separate script to handle txtsetup.sif
- Corrected the right-click menu ScriptFile option wich should output %ScriptFile%
- Fixed the extract command it's working fine for single files
- Corrected the right-click menu on the main window
- Most scripts were revised and cleaned from unnecessary files

#### [036] (2006-06-07)

- Fixed some registry problems when writing keys
- Fixed a fail when a project dir was listed but not found
- polished some icons
- Added the keyboard script to the standard distro
- Fixed the refresh button (to properly search all scripts in folders)
- Fixed the lock edit mode button in the source window

### **WinBuilder Projects**

A WinBuilder project is a collection of <u>scripts</u> that are typically used to build a specific boot disk. Specific instructions and/or details are normally included to help you specify a compatible source and how to debug some common issues that might sometimes appear.

You should always read the information page of each project you intend to use before you start building a project.

If you still have any doubts about how to proceed that are not covered here, you're invited to ask for help. For more on the currently supported projects, visit the **WinBuilder Project Forum** area.

### **Special project settings**

Please note that some projects may require specific conditions to be properly executed such as VistaPE which requires being used exclusively on NTFS partitions to properly handle WIM images. Please read the specific project page to view more details.

#### **Running your first project**

Click on the root item of each project to view how it should be used. In most projects you just need to click the Play button (blue triangle) and a wizard will ask you for the project source files location and continue the project build. Whenever you need to change this default location, you can click on the "Paths" tab and modify as needed.

Also, check to see if there are any <u>tutorials</u> for your chosen project by looking in the forums.

#### Creating a new project

You can create your own personal project as well by following these simple steps.

1. Create a folder for your project in the Winbuilder\Projects directory.

```
Example: C:\Winbuilder\Projects\MyProject
```

- 2. Create an empty file called script.project in this new directory.
- 3. Place the following code in the script.project file you just created.

```
[Main]
Title=My Project
Type=script
Author=Myself
Description=My New Project
Credits=
Version=1
Download_Level=0
Level=1
Contact=
Date=
Depend=
TargetDir=C:\Winbuilder\Target\MyProject
ISOfile=C:\Winbuilder\ISO\MyProject.iso
SourceDir=
```

4. Refresh your build tree by clicking the "Refresh" button or restarting Winbuilder.

Congratulations! You now have a skeleton project that you can start adding scripts to!

## **Application Scripts**

- Overview
  - Advantages
  - <u>Disadvantages</u>
- <u>Usage</u>
  - Settings on the [Main] section
  - Settings in the [variables] section
  - Settings in the [process ] section
- Frequently Asked Questions
- Bugs and Exceptions

### **Overview**

Application Scripts is how projects that use this method are able to create application scripts that are to be shared and used with different projects. These scripts are specifically designed to add programs to your projects.

#### **Advantages**

- Quickly allow you to write a new script that adds a program to a project with very few lines of code.
- Easy to learn with very intuitive functions
- Creates scripts compatible across different projects
- Long lasting scripts that don't need to be rewritten to work on newer projects that support this method
- Available on the Create Script tab only need to add your files and change the title, description and folder name

### **Disadvantages**

- Very limited scope of functions, only basic functions are supported.
- Targeted to smaller programs that need few files and settings.
- The target program needs to be freeware in most cases to be distributed with the script
- Still in development

### **Usage**

An application script isn't much different from "regular" scripts but it will take for granted that you follow a few definitions and rules and this way have enough conditions that are used by all compatible projects

Let's study a script example right now and start explaining how it works.

[main]
Title=Sudoku
Description=The popular number game
Level=5

[variables]
%ProgramTitle%=Sudoku
%ProgramEXE%=sudoku.exe
%ProgramFolder%=Sudoku

[process]
Add\_Shortcut, Desktop
Add\_Shortcut, AutoStart
Add\_Shortcut, StartMenu, Games and Fun
unpack

As you may notice - very few lines are needed, also note that this script is composed by 3 script sections: **Main** Variables **Process** 

### Settings on the [Main] section

More settings in this section can be added but only Title, Description and Level are needed as a minimum. Since this script is meant to add programs you need to use Level=5 value to group it with other similar scripts

#### Settings in the [variables] section

- %ProgramTitle% Used as title for the created shortcuts. We can use any description as desired, no need to be directly related to the exe or folder.
- %ProgramEXE% Program Executable used as the target application that is launched when clicking on the respective shortcut (if created), it can also be used to launch any other files besides executables like text files, html pages, bat files, etc.
- %ProgramFolder% The folder name where all files inside the script will be placed. You only need to add the folder name you wish to use, there's no need to add the full path.

#### Settings in the [process] section

This is the section that is executed when running the script so it's where we should place our functions.

Let's list and explain each one of the command functions used in the [process] section:

- 1. Add\_Shortcut, Desktop This function will add a shortcut on the desktop
- 2. Add\_Shortcut, AutoStart Mark your program to be included on the group of programs that runs on start up.
- 3. Add\_Shortcut, StartMenu, Games and Fun In this example, we are adding a shortcut on the start menu. Notice that applying the additional parameter as "Games and Fun" will place your shortcut inside a folder on the start menu with this name.
- 4. unpack A very important function because it will do most of the hard work of adding the needed files to your project. For more details, see the Common API command command reference page for the unpack command.

### **Frequently Asked Questions**

#### Q: How is this script used after all?

A: It is used to add simple programs and tools on your project in a quick and simple manner.

#### Q: What can be considered as "simple programs"?

A: In this category should be considered programs labeled as "standalone" and "portable" - which use very few files and registry settings. Also note that any program under this category should also be prepared to work under all available projects that support this method.

# Q: Well, I need a special icon different than the one used by the script - how can I do it?

A: Don't use this generic script for these cases, read the chapter about exceptions and how to handle them in your scripts

# Q: I want to use a script to add notepad on the windows directory instead of the programs folder, why can't these functions support this?

A: This is a specific feature that should be handled as an exception, you shouldn't use this method for these cases.

#### Q: How can we add more functions to the application scripts?

A: You will need to suggest and discuss them on the respective discussion

topic, in most cases do remember to ask functions that can also provided by other projects to make them globally available

# Q: Can I use this method and the project specific way to create these scripts together?

A: Yes you can, but it is not adviced. Both methods can work well together but should be kept in separate scripts. A script that was written with this common method should avoid using methods specific to a particular type of project.

#### Q: Where can these scripts be found or identified?

A: They can be included either inside your projects mixed with other scripts or you can find them in the download section of the community forums under the Application scripts category. We are also discussing a common folder where all these generic scripts could be placed but it's still in development.

#### Q: Is this method perfect?

A: No way! But we hope that this is a good method to create scripts that can work on most projects available at this moment and even on the ones to come in the future. This way we ensure that your favourite project can use as many available scripts as possible within the next years without having to rewrite or edit anything - making it easier for you to find more available programs.

# **Bugs and Exceptions**

You should look on the discussion forums for assistance in case you find a script that doesn't work as you expected so that the community can properly view it and suggest other alternative ways to solve exceptions.

### **Basic Script Template**

```
[main]
// Script title to be displayed in winbuilder
Title=My Program
// Short description of the program/script. This will be visible in
Description=An easy example script
// The script will be selected to run
Selected=True
// Run level 5 is used for Application Scripts.
Level=5
// Script Version
Version=1
// Author of the script - That's you!
Author=Homes32
// Make sure to give credit to anyone who helped you here! :)
Credits=Thanks to all who tested and gave feedback! YOU make this sc
// Date you released this version of the script.
Date=02-17-12
// Contact link - Ususally set to the forum topic you will use to su
Contact=http://reboot.pro/
// Pre defined variables go here
[Variables]
// Program title - Will be used for shortcut names
%ProgramTitle%=My Cool Program
// Main Program Exe. - Will be used for shortcuts
%ProgramEXE%=MyProgram.exe
// Folder where the program will live in PE
%ProgramFolder%="My Program"
[Process]
// Run from RAM will place the program files in WIM/RAMDRIVE where i
If, %ScrollBox_RunFromWhere%, Equal, "Run From RAM", RunFromRAM, True
If, %ScrollBox_RunFromWhere%, Equal, "Run From CD", RunFromCD, True
Echo, "Processing %ProgramTitle%..."
// Create our Directories
If, ExistDir, "%Target_prog%\%ProgramFolder%", DirDelete, "%Target_prog%
DirMake, "%Target_prog%\%ProgramFolder%"
// unpack our files from our archive we encoded into the script.
Unpack, Files, MyProgramArchive.7z, True
// Optionally we can add any files
```

```
// needed from windows such as the c runtime
// Generally you can skip this step unless your
// project does not provide the needed files.
Require File, crtdll.dll
// Process Registry values
// I you have a large number of registry
// entries it makes it easier to run them
// from a seperate section as we are doing here
Run, %ScriptFile%, Registry
// Add_Shortcuts
If,%CB_Desktop%, Equal, True, Add_Shortcut, Desktop
If,%CB_StartMenu%, Equal, True, Add_Shortcut, StartMenu,%IN_SMFolder%,
If, %CB_QuickLaunch%, Equal, True, Add_Shortcut, QuickLaunch
// Autorun the program with the /install parameter when PE starts
AddAutoRun, "My Program", "%PE Programs%\%ProgramFolder%\%ProgramEXE%"
[Registry]
// load the HKEY_USER hive and write our string value
RegHiveLoad, Tmp_Default, %RegDefault%
RegWrite, HKLM, 0x1, "Tmp_Default\Software\My Program", "MyKey", "MyValue
RegHiveUnLoad, Tmp_Default
RegHiveLoad, Tmp_Software, %RegSoftware%
[Interface]
CB_Desktop=Desktop, 1, 3, 21, 41, 110, 20, True
CB_StartMenu="Start Menu", 1, 3, 21, 81, 115, 20, True
CB_QuickLaunch=Quicklaunch, 1, 3, 21, 61, 110, 20, True
IN_SMFolder="Start menu folder:",1,0,36,121,100,20,AntiSpyware
pBevel1=pBevel1, 1, 12, 6, 11, 140, 145
pTextLabel1=Shortcuts, 1, 1, 16, 19, 75, 20, 8, Bold
ScrollBox_RunFromWhere="Run from RAM", 1, 4, 153, 11, 127, 21, "Run Default
```

### **Script Commands (By Category)**

The available scripting commands are grouped here according to their functionality. Usually all command which compare etc. are case insensitive. But, there are some commands having a "X" at the end. These commands work like that ones without the "X", but case sensitive.

- Attachments
- File Operations
- IF Clauses and Command Blocks
- INI Manipulation
- Miscellaneous Commands
- Network Tools
- Registry Handling
- Retrieve Functions
- String Formating
- Text Files

To access the list of all script commands in alphabetical order, see the **Script Commands** (Alphabetical) page

### **Attachments**

Attachments are files that can be included inside your script.

Since script files are meant to be edited by any text editor, these files are encoded into ascii text. This method allows you to add small files along with your scripts in order to keep everything together.

While a script is running you can use some of the following commands to best take advantage of this feature.

List of related commands

- ExtractFile
- ExtractAndRun
- ExtractAllFiles
- ExtractAllFilesIfNotExist
- Encode

### ExtractFile

Extract a single file from a folder inside a given script.

### **Syntax**

ExtractFile, <%ScriptFile%>, <Folder>, <Filename>, <DirToExtract>

#### **Parameters**

%scriptfile%	a variable that refers the filename of the script, it is used when you want to extract a file from the same script that you are running. You can also replace this variable with other filenames and extract files from them.
Folder	is a directory where the file is placed under the attachments structure which is similar to a very simple filesystem, this way you can add several folders inside a script to organize your files the way you prefer.
Filename	is the name of the file that you want to extract. No path is required, only the file name.
DirToExtract	is the target folder where you want to place this file.

### **Example:**

Using this example we will extract a given myFile.txt from myFile.script to the c:\ directory

ExtractFile,C:\myFile.script,Folder,myFile.txt,c:\

#### See Also

 $\underline{ExtractAllFiles}\ \underline{ExtractAllFilesIfNotExist}$ 

### **ExtractAndRun**

Extract a file that is attached (encoded) inside a script to a temporary location, run the file, and then delete it.

#### **Syntax**

ExtractAndRun, <%ScriptFile%>, <Folder>, <Filename>[, <parameters>]

Optional run parameters can be defined. They must not contain spaces, quotes or commas. Use the escaped form of these characters (i.e. #\$s, #\$q, #\$c) instead. It is useful when you only wish to run a file for a very specific purpose and this way avoid having to add extra commands to

- 1. extract the file
- 2. execute the file extracted
- 3. clean up (i.e. delete the extracted file)

### **Example:**

Using this syntax we will extract myApp.exe to a temporary location and then delete this file when it has finished running.

ExtractAndRun, C:\myFile.script, Folder, myApp.exe

### **ExtractAllFiles**

Extract all files from a given folder inside a script.

#### **Syntax**

ExtractAllFiles,<%ScriptFile%>,<Folder>,<DirToExtract>

Uses the same syntax as the <u>ExtractFile</u> command which can be used to extract a single file, but with the advantage that you can extract all files from a folder without needing to specify individual filenames.

### **Example**

Using this example we will extract all files found inside a folder in a given script file to c:\

ExtractAllFiles,C:\myFile.script,Folder,c:\

#### See Also

ExtractFile ExtractAllFiles ExtractAllFilesIfNotExist

### ExtractAllFilesIfNotExist

Extract all files from a given folder inside a script. Only files that don't exist in the target location will be extracted.

### **Syntax**

ExtractAllFilesIfNotExist,<%ScriptFile%>,<Folder>,<DirToExtract>

#### See Also

ExtractFile ExtractAllFiles

# **Encode**

Embed files inside a script.

### **Syntax**

Encode, <%ScriptFile%>, <Folder>, <Filenames>

#### **Remarks**

Winbuilder hs the ability to contain binary and other files inside the script files. These are stored in an ASCII encoded format so that they can be handled as text files.

You can either specify a single filename or use wildcards that match a file criteria. If the folder doesn't exist, it is created otherwise files are added to the already available folder.

### **Example:**

With this syntax we add c:\myApp.exe inside Folder to C:\myFile.script

Encode, C:\myFile.script, Folder, c:\myApp.exe

## **File Operations**

Commands under this category were designed to handle files and are particularly useful to create the base structure of your project or add additional features that you need.

#### List of available functions

- CopyOrExpand
- <u>DirCopy</u>
- <u>DirDelete</u>
- DirMove
- DirMake
- Expand
- FileCopy
- FileDelete
- FileRename
- FileCreateBlank
- <u>FileByteExtract</u>

# CopyOrExpand

Copy files from Install CD's where a file can be present either in normal or compressed format.

CopyOrExpand, <FileToCopyOrExpand>, <DestinationFolderOrFilename>[,PRE

#### **Remarks**

Compressed files often use the cabinet archiving method and usually have an identifiable character as the last character of their extensions. (e.g. myFile.Tx\_ ). Windows XP often uses this method to compress files on the install CD.

- If the destination file already exists, CopyOrExpand gives a warning in the log, and will overwrite the existing file.
- If the optional **PRESERVE** parameter added, CopyOrExpand *of a single file* will not overwrite an existing file *however* a warning is still given.
- The warning can be suppressed by an additional parameter **NOWARN**.
- The parameters **PRESERVE**, and **NOWARN** can be used simultaneously. The order does not matter.

**DestinationFolderOrFilename** means that you can either specify a folder where this file will be placed with the original (expanded) filename or new filename that you prefer.

#### **Example**

In this example CopyOrExpand will first try to find a file c:\MyPath\myTextFile.txt and if it is not found then it will look for c:\MyPath\myTextFile.tx\_ and expand this file if successfully found.

CopyOrExpand,c:\MyPath\myTextFile.txt,c:\MyPath\myNewerTextFile.txt

# **DirCopy**

This command will copy a folder and respective sub folders from a given source to a specified destination.

DirCopy, <FromDir>, <ToDir>[, SHOW]

#### **Remarks**

- DirCopy will overwrite any previously existing folder.
- The optional SHOW parameter will show the OS internal progress window.

#### **Example**

In this example DirCopy will copy the complete directory MyPath to AnotherPath

DirCopy,c:\MyPath,c:\AnotherPath

In this example DirCopy will copy all files and directories inside MyPath to AnotherPath

DirCopy,c:\MyPath\\*.\*,c:\AnotherPath

## **DirDelete**

Delete the entire contents of a directory. Deleted folders are not recoverable from the recycle bin.

DirDelete, <DirToDelete>[,FAST]

### **Example:**

DirDelete,c:\MyPath

#### **Remarks**

As of WB076, the optional **FAST** parameter was added. When this parameter is used, WinBuilder will rename the directory to a random name, and then execute the delete process in the background. This can improve performance when removing directories that have a large number of files in them.

• **Note:** This background process (i.e. the remove directory command) is a child of WinBuilder. When ( or **IF** ) WinBuilder is exited before the delete is finished, the background process is also terminated, and may leave a directory with a name like ~<???>. tmp left behind. If this happens, just manually delete this directory.

## **DirMove**

Move a folder and its contents from a given source to a specified destination overwriting any existing folder.

DirMove, <MoveDir>, <ToDir>

## **Syntax example:**

DirMove,c:\MyPath,c:\AnotherPath

# **DirMake**

This command will create a new folder.

DirMake, <NewDir>

### **Example:**

In this example, we create the directory called "MyPath" on the "C:" drive DirMake, c:\MyPath

# **Expand**

Decompress all files inside the specified cabinet (.cab) file to the folder you specify.

Expand, <CABfile>, <DirToDecompress>, <Optional Parameters>

A single file can be extracted from the cab file, by giving it's name as third parameter

Expand, c:\MyPath\MyFile.cab, c:\MyNewPath, YourFile.dll

#### **Remarks**

- Expand of a single file gives a warning in the log, if the destination file already exists.
- If the optional parameter PRESERVE added, Expand of a single file does not overwrite an existing file.
- A warning is still given

The warning can be suppressed by an additional parameter NOWARN

• The parameters PRESERVE, and NOWARN can be used simultaneously. The order does not matter.

#### **Example**

In this example Expand will try to expand all files found inside c:\MyPath\MyFile.cab to the folder c:\MyNewPath

Expand,c:\MyPath\MyFile.cab,c:\MyNewPath

# **FileCopy**

Copy source file(s) to a specified destination.

FileCopy, <FromFile>, <ToPath>[, PRESERVE][, NOWARN][, SHOW][, NOREC]

#### **Parameters**

1	FromFile	The file to copy. Wildcards (*.*) are allowed.
2	ToPath	The destination path. It may be just a directory, and if so, the file name will be preserved. If a filename is specified as a part of the destination, the file will be renamed. If the file exists in the target directory it will be overwritten and a warning will be written to the log.
-	PRESERVE (Optional)	Do not overwrite existing files. A warning will be written to the log.
_	NOWARN (Optional)	Do not write a warning message to the log if the file exists.
_	SHOW (Optional)	When using wildcards the progress bar shows single file processing, and the actual copied file name is shown.
_	NOREC (Optional)	When using wildcards only files in the initial folder will be copied, sub-directories will be ignored.

#### Remarks

- The optional parameters PRESERVE, NOWARN, SHOW and NOREC can be used simultaneously. The order does not matter.
- When using wildcards the warning mechanism is switched off.
- Trying to copy wildcard source to a single destination file raises an error.

#### **Example:**

In this example FileCopy will copy all text files (\*.txt) from MyPath and it's subdirectories to AnotherPath (using the C: drive)

FileCopy,c:\MyPath\\*.txt,c:\AnotherPath

# **FileDelete**

Delete the specified file(s).

FileDelete, <FileToDelete> [, Nowarn][, Norec]

#### **Parameters**

1	FileToDelete	The full path to the file(s) including sub-directories to delete. Wildcards are supported. If the file does not exist or no files match the wildcard a warning will be logged.
_	NOWARN (Optional)	Do not log a warning if the files don't exist
_	NOREC (Optional)	Only matching files in the specified directory will be deleted. Sub-directories will be ignored.

- The optional parameters Nowarn and Norec can be specified in any order.
- All deleted files are immediately removed. You cannot recover them from the recycle bin.

### Example 1:

In this example FileDelete will delete all text files (\*.txt) from c:\MyPath

FileDelete,c:\MyPath\\*.txt

#### Example 2:

FileDelete,c:\MyPath\\*.txt,NoWarn,Norec

In this example FileDelete will delete all text files (\*.txt) from the MyPath directory only. Any .txt files in any sub directories of MyPath will NOT be deleted. If no files are found a warning will not be added to the log file.

#### Note:

The Nowarn and Norec parameters were added in Winbuilder 082. They will not work with older versions.

# **FileRename**

Rename a single file.

FileRename, < OldFilename >, < NewFilename >

FileRename, c:\MyPath\myTextFile.txt, c:\MyPath\myOldTextFile.txt

## **FileCreateBlank**

Create a new empty file with the filename you specify. If a file with the same filename already exists it will be overwritten.

FileCreateBlank, <NewFile>[, PRESERVE][, NOWARN]

#### **Parameters**

1	NewFile	The full path of the empty files to create. If the file exists it will be truncated and a warning will be logged.
-	PRESERVE (Optional)	Do not over-write the file if it exists.
_	NOWARN (Optional)	Do not log a warning if the file exists or is over-written.

The parameters PRESERVE, and NOWARN can be used simultaneously. The order does not matter.

FileCreateBlank,c:\MyPath\MyNewFile.txt

# FileByteExtract

Extract binary data from a file matching the search criteria for a specific byte signature.

FileByteExtract,<FilesToSearch>,<OutputFile>,<ByteSignature>,<BytesL

#### **Remarks**

Byte signatures are specified in groups of two digits that are expressed in hexadecimal format. Possible value range from combinations starting at 00 and ending at FF

If this byte sequence is found, then it will copy a specific number of bytes and save them as a new file.

The bytes that compose the signature are also the header bytes on the new file.

This command was originally designed to find a byte sequence inside some system files that contained resources. Please look in the forums for practical examples of usage.

In this example, if a match is found then a new file will be created with the size specified in BytesLength

FileByteExtract,c:\MyPath\\*.\*,c:\MyPath\NewFile.bin,001122334455,144

#### **IF Clauses and Command Blocks**

If the evaluation of an If condition is 'True', the command following the condition is processed. If the evaluation of an If condition is 'False', the command of the Else command (if present), is processed. In both cases, instead of a command, there can also be a command block.

- If Clauses
- Begin / End Block
- Exist / NotExist
- Comparison Operations (Equal / NotEqual / Smaller / Bigger)
- Ping
- Online
- License
- Not
- Question
- ExistMacro
- ExistRegMulti

#### **If Clauses**

In the WinBuilder script language you can use the If statement to evaluate certain conditions and create conditional branches of execution. If a condition is true when you test it with the IF command then a command will be executed. The general format of this command is:

```
If, <condition>, <command>
```

Starting with WB076, the ability to have an alternate command executed was added by following the If statement with an optional else command. If a condition is false when you test it with the IF command then the command following the Else keyword will be executed.

• Note: The Else command must follow the If command immediately.

```
If, <Condition>, <Command>
Else, <Command>
```

In the following example we are testing if c:\myFile.txt does not exist. If this is true, then FileCreateBlank,c:\myFile.txt will be executed, otherwise a message windows with the text File already exists appears.

```
If, Not, ExistFile, C:\myFile.txt, FileCreateBlank, C:\myFile.txt
Else, Message, "File already exists", INFORMATION
```

This is a very powerful construct, since the <command> specified can be **ANY** valid script command. This was further enhanced with the addition of command blocks using Begin / End, which allows multiple commands to be executed.

## Begin / End

These commands are used to mark the start and end of a command block and are used with the conditional statements (<u>if `IF / ELSE`</u>) to form a command block. This is used when multiple commands are desired to be executed as the result of a single conditional test.

```
Begin
<command>
<command>
...
<command>
End
```

Here the individual commands can be any valid script command. Because <command> can also be an If, nested command blocks are possible.

```
If,%OSName%, Equal, xp, Begin
    If,%SP%, NotEqual, sp3, Begin
        If,%pCheckBox1%, Equal, True, Run,%ScriptFile%, Process-KB893803
    End
    Else, Begin
        Set,%hotfix%, 1
        Run,%ScriptFile%, Process-Copy
    End
End
Else,...
```

#### Exist...

You can use these tests to determine if a file, directory, script section, or variable does or does not exist. If this test is true, then a Command is executed. Based on the specific test selected, one or more additional arguments may be needed. Avaiable tests include:

ExistFile	Not,ExistFile	Check to see if a file does/doesn't exist <sup>1</sup>
ExistDir	Not,ExistDir	Check to see if a directory does/doesn't exist <sup>1</sup>
ExistSection	Not,ExistSection	Check to see if a given section does/doesn't exist inside the specified file
ExistRegSection	Not,ExistRegSection	Check to see if a given section does/doesn't exist in the specified registry
ExistRegKey	Not,ExistRegKey	Check to see if a specific key does/doesn't exist inside the specified registry
ExistVar	Not,ExistVar	Check to see if a specific variable does/doesn't exist - i.e. has it been defined

1. Wildcards may *only* be used in the File and Directory existence checks. *BUT*, they may not give expected results. For example, checking to see if \*.\* exists *will ALWAYS return TRUE* because of the "dot" and "dot-dot" entries returned by the internal API used.

If, EXISTFILE, TestFile, Command
If, EXISTDIR, Directory, Command
If, Not, EXISTSECTION, TestFile, Section, Command
If, EXISTVAR, %Var%, Command

In the example below we are testing if c:\myFile.txt does not exist. If this condition is true, then FileCreateBlank, c:\myFile.txt will be executed

If, Not, ExistFile, c:\myFile.txt, FileCreateBlank, c:\myFile.txt

The following command will read the value of HKLM\SOFTWARE\Classes\xyz, Language, if RegSection HKLM\SOFTWARE\Classes\xyz does exist.

 $If, \verb|ExistRegSection|, \verb|HKLM|, \verb|SOFTWARE| \verb|Classes| \verb|xyz|, \verb|RegRead|, \verb|HKLM|, \verb|SOFTWARE| \verb|Classes|, \verb|xyz|, \verb|RegRead|, \verb|HKLM|, \verb|SOFTWARE|, \verb|Classes|, \verb|xyz|, \verb|RegRead|, \verb|HKLM|, \verb|SOFTWARE|, \verb|Classes|, \verb|xyz|, \verb|RegRead|, \verb|HKLM|, \verb|SOFTWARE|, \verb|Classes|, \verb|xyz|, \verb|RegRead|, \end{tabular}, \end{tabular}$ 

This command will read the value of HKLM\SOFTWARE\Classes\xyz, Language, but only if the registry key HKLM\SOFTWARE\Classes\xyz, Language exists.

If, ExistRegKey, HKLM, SOFTWARE\Classes\xyz, Language, RegRead, HKLM, SOFTW

In this final example we will check if the program path C:\Program Files\IrfanView exists. If it does, then the value of %Var% will be set to 1. Then we will check to see if %Var% exists. If it is true (i.e. was created above or even earlier in the script or project) a message windows with the text "1 does exist" appears.

Set,%var%,NIL
If,ExistDir,C:\Program Files\IrfanView,Set,%Var%,1
If,ExistVar,%var%,Message,"'%var%' does exist",Information

# Comparison Operations for If Equal / NotEqual / Smaller / Bigger

This set of IF clauses can compare numeric values as well as text strings.

If, <%var%>, <test>, <value>, <Command>

For comparing text strings, WB078 added EqualX which does a case sensitive comparison.

In this given example we are testing if %myVariable% is equal to 10. If this condition is true then FileCreateBlank, c:\myFile.txt will be executed

If,%myVariable%,Equal,10,FileCreateBlank,c:\myFile.txt

# **Ping**

Ping will test if a given IP is valid and run a command if this test is true.

If,ping,<xxx.xxx.xxx.xxx>,<Command>

In this given example we are pinging a machine on the network. If this condition is true then FileCreateBlank, c:\myFile.txt will be executed. WB078 added the ability to use a hosts domain name in addition to the IP address.

```
If,ping,10.111.8.1,FileCreateBlank,c:\myFile.txt
If,ping,winbuilder.net,FileCreateBlank,c:\myFile.txt
```

# **Online**

This command will test if the computer is online and run a command if this test is true.

If, online, <Command>

If, online, FileCreateBlank, c:\myFile.txt

## License

This command was added to be used as tool to accept a license agreement whenever needed. It will open a special tab where the user has the option to read the terms of the license and then press a button corresponding to his decision to agree or not with the displayed terms.

If, License, <textfilename>, <Command>

Here, TextFilename is the text file that will be displayed on the license window.

The use of License runs the command when the user agrees with the license conditions and Not, License will cause the command to run if the user doesn't agree.

This example will present the contents of myEula.txt - in case the user doesn't agree with this license the project build will be aborted.

If,Not,License,c:\myEULA.txt,halt,"User doesn't agree with license."

## Not

This syntax was introduced to simplify the addition of future conditions. It basically changes the "sense" of the check from true to false. In other words, the statement If, Not, %var1%, EQUAL, %var2%, command is equivalent to If, %var1%, NOTEQUAL, %var2%, command.

If, Not, <Condition>, <Command>

#### **Note:**

Many of the earlier conditional tests had an explicit form of negative test. That syntax is still supported for compatibility, however, all tests of the form <notXxxxxxx> are internally converted to <not, Xxxxxx> before being processed by WinBuilder

If,Not,%var1%,EQUAL,%var2%,FileCreateBlank,C:\myFile.txt

# Question

Asks the user a question, and uses their response to determine if the command should be executed. Note: all strings defined in the language files can be used like variables, identified by their ID number \$#nnnn.

If,QUESTION,<Question>,[<timeout>,<True/False>]<Command>

#### **Remarks**

Usually the question dialog waits for the user's input. Using these optional parameters, you can cause to close it after a timeout (and supply a default response), by adding the timeout (in seconds) and a default Value (True/False) as optional arguments before the command. If they are specified, **BOTH** the timeout value (seconds) and default value when the timeout occurs must be specified.

If,QUESTION,"Exit Script?",10,TRUE,Exit,"Exit selected"

If, QUESTION, \$#1541?, command

Here, we are using an internal string to display a question (The question mark is part of the TEXT, not part of the ID!). The resulting dialog box shows in case of default language (English):

Save changes to disk?

In case of e.g. italian language:

Salva cambiamenti al disco?

#### ExistMacro

The command checks whether the macro is defined in the [Variables] section.

#### **Syntax**

If,ExistMacro,<Macro>,<Command>

#### Note

This command was added in Winbuilder 80; It will **not** work with older versions.

### ${\bf Exist Reg Multi}$

The command checks for the existence of a substring in a multiple string

#### **Syntax**

If,ExistRegMulti,<HKEY>,<Section>,<Key>,<SubString>,<Command>

#### **Example:**

If,EXISTREGMULTI,HKLM,SYSTEM\ControlSet001\Control\ServiceGroupOrder

#### Note

This command was added in Winbuilder 80; It will not work with older versions.

### **INI Manipulation**

This category contains commands that were created to work with INI based text files. INI files group information using a few rules, it is also very easy to understand and edit when needed.

We use INI files to exchange information between scripts or store data whenever needed. In most cases INI language files use extension .ini - but our commands are not limited to just this type of extension and can be used on any text file that uses a INI derivated language like .script language.

#### Quick explaintion of some INI concepts:

- A group of data (section) is always contained between [] (e.g. [MySection] ) and will only finish whenever another section is found or when the file has no more lines to read.
- Each entry has a Title and is followed by it's respective data. Both are separated by an equal sign (e.g. Color=Blue)

#### Available INI functions

- IniWrite
- IniRead
- IniDelete
- IniAddSection
- IniDeleteSection
- IniWriteTextLine
- <u>IniMerge</u>

## **IniWrite**

#### **Syntax**

IniWrite,<Filename>,<Section>,<Key>,<Value>

This command will write the key and value on the section inside the file you specify.

In this example the specified myKey and respective myValue will be written in the section mySection in C:\myFile.ini

IniWrite,C:\myFile.ini,mySection,myKey,myValue

## **IniRead**

Read the value of a key from a section inside a file that you specify.

IniRead,<Filename>,<Section>,<Key>,<%variable%>

In this example the value of the specified key will be stored inside %myVariable%

IniRead, C:\myFile.ini, mySection, myKey, %myVariable%

## **IniDelete**

Delete the specified key from a section inside a file.

IniDelete, <Filename>, <Section>, <Key>

### example:

In this example the specified key and respective value will both be deleted IniDelete, C:\myFile.ini, mySection, myKey

### **IniAddSection**

This command will add a new section inside the file you select.

If this section already exists no action will take place, otherwise it will create a new empty section in the respective text file.

IniAddSection, <Filename>, <Section>

### example:

IniAddSection, C:\myFile.ini, mySection

## **IniDeleteSection**

Remove a given section and its contents inside the file you select.

IniDeleteSection, <Filename>, <Section>

In this example we're deleting an existing section.

IniDeleteSection, C:\myFile.ini, mySection

## **IniWriteTextLine**

This command will add a text line to given section inside the file you select.

IniWriteTextLine,<Filename>,<Section>,<StringLine>[,APPEND]

By default new lines are placed on the top of the section. To place new lines at the end of the section use the APPEND parameter.

In this example a new line is added on the top of the section and all remaining lines are moved below.

IniWriteTextLine, C:\myFile.ini, mySection, ThisIsMyLine

Optional APPEND argument so that the new line is added on the end of the section

IniWriteTextLine, C:\myFile.ini, mySection, ThisIsMyLine, APPEND

# IniMerge

Update all values from a INI file based on the contents from another INI file.

IniMerge,<UpdatedINIfile>,<INIfileToUpdate>

All Sections are compared inside both files and newer keys or items with different values are added to INIfileToUpdate

In this example the command will read all keys found inside myNewValues.ini and if they aren't present, ThisIsMyBaseFile.ini will be added, overwriting older values if necessary.

IniMerge, C:\myNewValues.ini, mySection, ThisIsMyBaseFile.ini

#### **Miscellaneous Commands**

Under this category you find the commands which aren't grouped under any specific category but are still quite used on most scripting tasks.

- Addvariables adds all variables listed inside a section
- Beep uses the system speaker
- Echo outputs a message (viewable on the log)
- <u>Exit</u> stops processing current script
- GetParam get the parameters passed to a script section
- <u>Halt</u> stops building the whole project
- Loop provides a simple looping construct
- Message outputs a message window
- PackParam pack the parameters to be passed to another section
- Run / Exec runs a section from a script file
- Set changes the value of a variable
- ShellExecute launches an external file
  - ShellExecuteDelete
  - ShellExecuteEx
- <u>System</u>
  - System,Comp80
  - System, Cursor
  - System, ErrorOff
  - System, FileRedirect 64bit systems only
  - System, GetEnv
  - System, GetFreeDrive
  - <u>System,GetFreeSpace</u>
  - System, HasUAC
  - System, Is Admin
  - System, Is Terminal
  - System,Log
  - System,OnBuildExit
  - System, On Script Exit
  - System, Rebuild Vars
  - System, RefreshInterface
  - System, RegRedirect 64bit systems only

- System,ReScanScripts
   System,SaveLog
   System,SplitParameters
   Wait "sleeps" the script for specified time

### **AddVariables**

Import the contents of the [Variables] section from another script into the current script.

AddVariables,<%ScriptFile%>,<Section>,[GLOBAL]

#### **Tips**

- When a script is started it will read the [variables] section, but using this command during the script processing, you can repeat this procedure and read the variables found inside any section you wish to use.
- Use %scriptfile% to read a section from the same file from where this command is called.
- Older variables with the same designation are overwritten with newer values.
- If the optional parameter ( **GLOBAL**) is specified, the additional variables will be known through the complete build.

AddVariables,%scriptFile%,AlternativeVariables

# Beep

Make some noise to draw the attention of the user.

Beep, <Action>

The selected **Action** can be one of the following values

- Ok
- Error
- Asterisk
- Confirmation

The sounds produced by each **Action** will vary based on the sound profile of the users computer.

beep,ok

## **Echo**

Display a message on the processing window while the script is running. This message will only disappear when another message is displayed or the script finishes.

echo,<message>

echo,"Hello World!"

## **Exit**

Force the current script to terminate and continue processing with the next script.

Exit, <exit message>[, NOWARN]

The "Exit message" is a small text message that is displayed on the log where you can specify the reason why Exit was called.

A warning icon is associated with the log entry but in some conditions, where the exit is intended and a warning icon would confuse the user the NOWARN parameter can be specified to associate an information icon instead.

Exit,"File was not found"

# **Getparam**

Writes the parameter at the given index position into **%var%**. This is needed when dealing with more than 9 passed parameters using the <u>Run or Exec</u> commands.

GetParam, Index,%Var%

### **Parameters**

1	Index	The position of the passed argument to read
2	%Variable%	The Variable to write the value to

#### **Remarks**

Parameters are accessible by only a single digit following the "#" (hash mark) character. Given the "fixed formatting", a different process was needed to access the values of any greater parameter.

This is also an easy way to make script files more readable, since it allows you to assign a more meaningful variable name to the parameters being passed into the section.

When running the command:

Run, %ScriptFile%, Animals, Cow, Dog, Cat, Goat, Donkey, Wombat, Turkey, Horse

[Animals]
GetParam, 10, %aBigAnimal%
Echo, %aBigAnimal%

The value of **%aBigAnimal%** will be Hippopotamus

# Halt

Force the overall project building to terminate.

Halt,<reason>

### **Parameters**

1 Reason A small text message that will be displayed in the log where you can explain why the build was canceled.

Halt, "Incompatible Source"

## Loop

Repeatedly execute a particular section in a script with a counter value that increments each time through the loop between the specified start and end values. Any additional parameters are passed to the script section.

Loop, %ScriptFile%, <Section>, <start>, <end>, <optional parameters.

The loop can be prematurely exited by using the following command

Loop, BREAK

This code is searching in all oem#.inf files in path C:\Windows\inf (oem0.inf up to oem100.inf) for an entry VBoxUSB.sys in section [SourceDisksFiles]. If it is found the oem.inf file name is set as result and the loop ends.

• NOTE: oem.inf files are always beginning with ZERO (oem0.inf)

```
Set,%Result%,?
Loop,%ScriptFile%,Try-OEM,0,100,SourceDisksFiles,VBoxUSB.sys
Echo,%Result%

[Try-OEM]
Set,%file%,C:\Windows\inf\oem#c.inf
If,ExistFile,%file%,IniRead,%file%,#1,#2,%Var%
If,-%var%,NotEqual,-,Set,%Result%,%file%
If,%Result%,NotEqual,?,Loop,BREAK
```

The Loop statement can increment characters as well as numbers, e.g. drive letters. In this example the Loop statement is looking for a specific file on all drive letters from a: to z:. If the file is found it will be executed.

```
Set,%a_searchFile%,"Program Files\IrfanView\i_view32.exe"
Loop,%ScriptFile%,Try,a,z
If,EXISTFILE,%pstart%,ShellExecute,OPEN,%pstart%

[try]
Set,%pstart%,#c:\%a_searchFile%
If,EXISTFILE,%pstart%,Loop,BREAK
```

# Message

Displays a simple message box with optional timeout.

Message, <message>, <Action>[, Timeout]

### **Parameters**

1	Message	Text to display to the user
2	Action	Defines the icon to display. It must be one of the following values
		Information - Displays a Information Icon
		Confirmation - Displays a Question Icon
		Error - Displays a Error Icon
		Warning - Displays a Warning Icon
3	Timeout (Optional)	Time in seconds before the window automatically closes.

Icons displayed may vary depending on your operating system configuration.

message, "Script has been completed!", Information, 10

## **PackParam**

Packs parameters into a user defined variable, each enclosed in quotes into a comma separated list.

PackParam, Start, %Var%[, %Num%]

This command packs all parameters beginning with the parameter in position <start> into %var%, each enclosed in quotes, and separated by comma, like:

```
"param4", "param5", . . .
```

PackParam optionally has an additional argument ( **%num%** ). It writes the number of arguments into the variable supplied. **%Num%** is always the number of ALL arguments, independent from where the packing process has started. The parameters can be declared as "OUT" variables, that means when returning from "Run", they will be updated with the actual value.

### Run / Exec

#### **Syntax**

Run, %ScriptFile%, <Section>[, parameters]
Exec, %ScriptFile%, <Section>[, parameters]

The **Run** command is used to process the commands found in a named section of a script file. You can run sections from any script including the script from where run is originally called.

Note: use the variable %scriptFile% as a parameter to call another section found inside the same script.

The Run command command will use the same variables available in the original script from where it was called. The Exec command has a very similar behavior. However, it will additionally add all the variables found inside the [variables] section of the script file selected. Identical variables found on the new script will overwrite the existing value.

This command also allows you to use parameters - this is a very useful feature to create a section that works based on these variables and this way avoid the need to add very long similar lines of code. In fact, this is used heavily by the common API implementation.

Inside each section that is called by the Run command, you can use the respective parameters - they are numbered from #1 to #9. This means that you can use a command like

Run, %scriptfile%, mysection, "Hello World!"

and inside the [mysection] of the same script you could use a command like echo, #1 to display "Hello World!" on your log. All variables are "passed by value". That means that they are usually read only. If you want to return an updated variable after the section has been completed, prefix it with OUT: to indicate that it should be updated. **Note:The "OUT" mechanism is not supported by EXEC.** 

If a 'Run' has more than 9 parameters, they can be accessed using two new commands added in  $WB078\,$ 

- GetParamPackParam

In this example we run a section called mySection inside the same script run, %scriptfile%, mySection

In this example, after returning from the "Run", %first% and %third% are unchanged, regardless whether inside the "Run" #1 and / or #3 are changed. %second% will get the actual value of #2 when "Run" is finished.

Run,%scriptfile%,mySection,%first%,OUT:%second%,%third%

#### Set

#### **Syntax**

Set,<%Variable%>,<NewValue>[,GLOBAL / PERMANENT]

With this command we can change the value of a variable. If more variables are linked to this variable, then you might need to call System, RefreshVars to ensure that all variables get refreshed values and work properly.

**WB076** added a new variable handling mechanisim that makes System, RefreshVars unnecessary.

Usually the variable's definition has a lifetime of the current script.

- You can make it valid for the rest of the actual build by adding the optional parameter GLOBAL
- You can make it valid for the lifetime of the project (i.e. write it's value into script.projects) by adding the optional parameter PERMANENT.

These aparameters are mutually exclusive.

Set,%workdir%,c:\myPath,GLOBAL

### **ShellExecute**

#### **Syntax**

ShellExecute, <Action>, <Filename>[, Parameters][, StartDir], [%ExitCode%

This function will launch a file using the settings you specify. WinBuilder will wait (pause) for the launched file or application to be closed before continuing to process the remaining commands.

- **Action** is the specific method you wish to use to start the external file. If not specified, the default "Open" Will be used. Possible action values are:
  - Open will open the file using the default associated application. This
    means that if it is an executable, the program will be started, and
    possibly open a console window to display it's output.
  - **Hide** will also open the file, but it will be launched in hidden mode (i.e. any console output will not be viewable)
  - Print print a file using the default printer
  - Explore open an explorer window (used to display the contents of folders)
- **Filename** is the complete filename to where the file can be found, you can omit the dir part if this file can be found on the system paths.
- **Parameters** (optional) are the set of switches that you can pass along as command line parameters when executing the file.
- **StartDir** (optional) is the location that you wish to specify as your default work directory.
- **%ExitCode%** (optional) is the name of the variable that will be updated with the exit value returned by any executable started. This can be useful to validate the exit code value returned by the program. This variable must already exist, or a warning will be given, and WinBuilder will **not** be able to capture the return code.

In this example we can open a new console window (cmd.exe) ShellExecute, open, cmd.exe

#### **See Also**

- <u>ShellExecuteDelete</u> has a similar behavior but will delete the application after it has been finished.
- <u>ShellExecuteEx</u> has a similar behavior but won't wait for the application to be finished and will continue with the script processing right after launching the target filename. Since it does not wait for any executable to finish, no %ExitCode% will be returned.

## **System**

System, < Action>

This command is used to perform some program internal actions and refresh values that were modified during the script execution. It can also set a flag that reminds the program to perform the specified action once the project is completed. <action> can have of the following values

- <u>Comp80</u>
- Cursor
- ErrorOff
- FileRedirect 64bit systems only
- GetEnv
- GetFreeDrive
- GetFreeSpace
- HasUAC
- IsAdmin
- <u>IsTerminal</u>
- Log
- OnBuildExit
- OnScriptExit
- RebuildVars
- RefreshInterface
- RegRedirect 64bit systems only
- ReScanScripts
- SaveLog
- SplitParameters

# System, Comp80

To improve processing speed, the way variables are treated internally was changed. In some situations, this may cause incompatible results. This setting allows the project to force compatible processing to occur.

System, Comp80, {On|Off}

#### **Remarks**

- By using System, Comp80, OFF you can make the current WinBuilder script processing (mostly) compatible to the processing of WinBuilder 080.
- When compatibility is set to off, scripts are processed 2 to 3 times faster.
- By default the compatibility is switched off (starting with version 082).

In this example, we want to go back to the slower, but compatible way of processing variables.

System, Comp80, On

# System, Cursor

Sets the system cursor as hourglass or the default cursor.

System, CURSOR, {WAIT|NORMAL}

# System, Error Off

This command allows a script processing error to not be marked as an error and therefore !Winbuilder will continue to process the scripts. By default this functionality is **only** valid for the next script line. That can be changed by defining the optional number of lines.

System, ERROROFF[, lines]

#### **Example:**

System, ERROROFF

RegRead, HKLM, "CurrentControlSet\Services\Eventlog\Application\VMware If, NOT, EXISTVAR, %VMtmp%, Begin

System, ERROROFF, 2

RegRead, HKCR, Wow6432Node\Applications\vmware-mount.exe\shell\Mount\c If, NOT, EXISTVAR, %VMtmp%, RegRead, HKCR, Applications\vmware-mount.exe\s

# System,FileRedirect

Redirect some of the OS directory calls to avoid possible conflicts.

System,FILEREDIRECT,{ON|OFF}

#### Remarks

Only necessary when WinBuilder is running on a 64bit system. On a 32bit system, this command has no effect.

#### **Example:**

System, FILEREDIRECT, OFF
ShellExecute, Hide, "cmd.exe", "/c imdisk -d -m %FreeLetter%"
System, FILEREDIRECT, ON

## System, GetEnv

Gets the value of a **system** Environment(a.k.a.ENV) variable -

System, GETENV, <EnvironmentVariable>,%var%

#### Remarks

The name of the <EnvironmentVariable> *must not* be enclosed in percent signs (%).

## **Example:**

System, GETENV, Temp, %var%
Echo, %var%

## System, GetFreeDrive

This command returns the highest available free drive letter, Usually the "Z:" drive

System, GetFreeDrive, <%var%>

## System, GetFreeSpace

Returns the free space of Path's drive in megabytes

System, GETFREESPACE, <Path>, <%var%>

This command computes the free disk space on the drive specifice, and puts result into %var%. Note: megaybtes = free bytes / (1024 **1024**)

#### **Example:**

System, GETFREESPACE, %BaseDir%, %var% System, GETFREESPACE, C:, %var% System, GETFREESPACE, C, %var% Result:

## System, HasUAC

This command will set the variable to true if Vista / Win 7 UAC ( User Account Control ) is enabled.

System, HasUAC, <%var%>

### Remarks

This is necessary, since UAC can cause problems while building some projects.

#### **Example:**

System, HasUAC, %var% If, %var%, Equal, True, begin

Message, "Error: UAC is enabled. This may cause problems when bui Halt, "Error: UAC is anabled. This may cause problems when buildi End

## System, Is Admin

Check to see if WinBuilder was started by an account with "Admin" privileges.

System, ISADMIN, %var%

Sets %var% to True or False, depending on whether WinBuilder has been started by an admin.

# System,IsTerminal

Check to see if !WinBUilder has been started on a remote desktop

System,IsTerminal,<%var%>

This command will set the variable to true if winbuilder is being run under terminal server ( remote desktop ). This can cause problems to some projects.

## **Example:**

System, IsTerminal, %var% If, %var%, Equal, True, begin

Message, "Error: It seems you are working in a Terminalserver ses Echo, "Error: It seems you are working in a Terminalserver session End

# System,Log

Control the output log functionality

System, Log, {on|off}

When a project needs to use a certain function many, many times, it could expand inside the log.html into many unnecessary lines. This command stops / restarts logging to log.html. It is most useful inside <u>API functions</u>.

It is only in effect for the remainder of the current script!

## **Example:**

System, Log, off

## System, On Build Exit

Specify the command to be executed after the last script in the project has been processed.

System, OnBuildEXIT, Command

This statement can be written anywhere inside the running part of any script. The last executed statement is the one that will be used when the last script has completed. This means different scripts can define different actions or deactivate this process (with System, ONBUILDEXIT - i.e. no command specified).

For more examples, see System, ONSCRIPTEXIT, which has a similar function.

## System, On Script Exit

Specify the command to be executed after the last statement in the current script has been executed.

System, ONSCRIPTEXIT, <Command>

This statement can be written anywhere inside the running part of the script.

If the command is 'Run', the 'type' of the exit is passed as parameter #1, and can be one of the following values:

- DONE normal finish
- STOP the user clicked the STOP button
- ERROR the script has been finished because of an error
- COMMAND the script has been finished because of a System,Halt or System,Exit command
- EXCEPTION a system exception occurred during process, e.g. 'index out of bounds'

#### **Example:**

```
System,ONSCRIPTEXIT, Message, "Everything Done!",INFORMATION
System,ONSCRIPTEXIT,Run,%ScriptFile%,MYEXIT
// force an error:
FileCopy,y,x

[MYEXIT]
If,#1,EQUAL,ERROR,Beep,ERROR
If,#1,EQUAL,STOP,Beep,WARNING
If,#1,EQUAL,DONE,Beep,OK
If,#1,EQUAL,COMMAND,Beep,CONFIRMATION
```

# System, Rebuild Vars

Rebuild the system variable table

System, REBUILDVARS

This is the successor of System, RefreshVars, which was deprecated as of WB076. In most cases it is no longer necessary (**or supported**).

#### **Example:**

```
Set,%First%, first
Set,%Result%,"Here is %first%"
Echo,Value of #$pResult#$p is: %Result%
// >> Here is first
Set,%First%, second
Echo,Value of #$pResult#$p is: %Result%
// >> Here is first
System,RebuildVars
Echo,Value of #$pResult#$p is: %Result%
// >> Here is second
```

# System, Refresh Interface

Refresh the **Graphical Interface** 

System, RefreshInterface

Mostly used as part of the code assigned to buttons or radio button selections to refresh the script interface with changes that may have occurred. Can also be used as part of making complex interfaces that change based on the values selected.

## System, RegRedirect

Redirect some of the OS call related to registry manipulation to avoid possible conflicts.

System, REGREDIRECT, {64|32|0FF}

### **Parameters**

- 64 causes next accesses to the 64 bit entries.
- 32 causes next accesses to the 32 bit node.
- OFF causes Host OS to make the decision

#### **Remarks**

- Maybe necessary when WinBuilder is running on a 64bit system.
- Once set, the definition is valid for the rest of the projects processing.
- Using this command on a 32bit system has no effect.

#### **Example:**

System, REGREDIRECT, 64
RegRead, HKLM, SOFTWARE\Sun\xVM VirtualBox", InstallDir, %VBPath%
System, REGREDIRECT, OFF

# System, ReScanScripts

Find and categorize scripts displayed on the main window.

System, ReScanScripts

This will repeat the procedure to find and categorize scripts displayed on the main window. Useful for cases when new custom scripts are created while building a project.

# System, SaveLog

Save the current accumulated log file

System, SAVELOG, <OutputFile>

Saves the actual accumulated log to <OutputFile>. <OutputFile> should have the extention '.html' (since !Winbuilder writes out the log in html syntax for easy viewing in a web browser - i.e. uses color coding as well as internal page links to provide navigation.

Usefull for debugging purposes - i.e. When a build creates a reproducible hang in a certain script, you can save the log before or in the beginning of this script.

# System, Split Parameters

Splits in a 'Run' or 'Exec' command a single parameter which contains commas, into several parameters.

System, SPLITPARAMETERS, {ON|OFF}

The System, SPLITPARAMETERS, ON command can be given once in the project, e.g. in the [Process] section of script.project, and is then valid for the complete project (or until it is switched off by a System, SPLITPARAMETERS, OFF command. It can then be switched on again later, and off, and so on ....)

## **Example:**

[process]
Set,%List%,"a,b,c"
Run,%ScriptFile%,Test,%List%
System,SPLITPARAMETERS,ON
Run,%ScriptFile%,Test,%List%
System,SPLITPARAMETERS,OFF

[Test] Echo,#1 Echo,#2 Echo,#3

This creates a log file result:

```
[Set, %List%, "a, b, c"]
SetVar - Sucessfuly changed the value of [%List%] to: [a,b,c]
[Run, %ScriptFile%, Test, %List%]
Run - Processing section: [Test] From file: [%BaseDir%\Projects\Scott\splitparameters.script]
with parameters: [a,b,c]
[Echo,#1]
i a,b,c
[Echo,#2]
1)
[Echo,#3]
1)
Run - Processed section [Test] in file: [%BaseDir%\Projects\Scott\splitparameters.script]
[System, SPLITPARAMETERS, ON]
System,SplitParameters,ON
[Run, %ScriptFile%, Test, %List%]
Run - Processing section: [Test] From file: [%BaseDir%\Projects\Scott\splitparameters.script]
with parameters: [a,b,c]
[Echo,#1]
i a,b,c
[Echo,#2]
i)
[Echo,#3]
Run - Processed section [Test] in file: [%BaseDir%\Projects\Scott\splitparameters.script]
[System, SPLITPARAMETERS, OFF]
System, SplitParameters, OFF
My Program
Script time: 151 milliseconds
Script has been sucessfully processed!
```

# **Visible**

Set the visibility of a UI element.

Visible, <variable to change>, True/False[, Permanent]

This is a more simplistic command than fully controlling all the parameters of the UI, but can be useful to customize the UI based on the state of a checkbox or a radio button setting. The optional parameter sets this new value into the file, so it will be permanent.

### **Example**

```
[Process_Visible]
Echo, "Please Wait"
// When we change state of the scroll box, then change
// visibility of label and folder box...
If,%ProjectSelectScrollBox%, Equal, "Visible - True", Begin
    Set,%Import_Custom_State%, True, Permanent
    Visible,%Import_Custom_Dir%, True, Permanent
    Visible,%Custom_Label%, True, Permanent
End
Else, Begin
    Set,%Import_Custom_State%, False, Permanent
    Visible,%Import_Custom_Dir%, False, Permanent
    Visible,%Custom_Label%, False, Permanent
    Visible,%Custom_Label%, False, Permanent
End
```

# Wait

## **Syntax**

wait,<seconds>

Using this command will pause the script processing the value of seconds that you specify. After this period is concluded, the process will continue.

# **Example:**

This example will make the application wait 5 seconds before continuing wait, 5

# **Network Tools**

Functions on this category allow you to work with the network environment whenever available.

- WebGet
- WebGetIfNotExist

## WebGet

#### **Syntax**

WebGet, <FileToDownload>, <FileOnDisk>[, MD5][, Ask][, Timeout]]

Use this function to download files from a network server to the local machine.

- It is assumed that both machines (local and remote) are prepared for this operation.
- No checks are done to ensure that the local machine is online or that is has enough disk space to download the file.
- The remote server must also be accessible and the file must be available for download.
- Only direct download links are accepted, dynamic internet pages are not valid.

MD5 check is an additional parameter that can be added to ensure that your file matches a security check based on the MD5 hash calculation.

- This is an optional parameter, only needed for cases when it is important to only accept a file that passes on this download integrity test.
- If a file fails this test it won't be copied over to the final destination as FileOnDisk.

The optional parameters of **Ask** and **Timeout** are only used *if enabled*, and if enabled, will display a question dialog box, and wait for a response. This can be used to let the user skip over a download operation.

- To use this feature, the line "Ask before WebGet" in the Winbuilder Tab TOOLS\OPTIONS has to be checked.
- To use optional timeout, the line "Use Timeout when asking" in the Winbuilder Tab TOOLS\OPTIONS has to be checked.

If the target filename folder doesn't exist, it will be created. If a file already exists it will be overwritten (read <a href="WebGetIfnotExist">WebGetIfnotExist</a> for alternative behavior)

### **Example:**

In this example we download the file iview423.zip from server foo.com to the folder  $c:\TrfanView\iview423.zip$ 

- only if MD5sum is fd490d07cc236cb44ef29abfd9daf13d
- with a question dialog's text string displaying 'ZIP-file'
- and automatically accept the dialog (i.e. timeout) after 10 seconds

WebGet,http://foo.com/iview423.zip,c:\IrfanView\iview423.zip,fd490d0

## WebGetIfNotExist

#### **Syntax**

WebGetIfNotExist,<FileToDownload>,<FileOnDisk>[,MD5][,Ask][,Timeout]

Use this function to download files from a network server to the local machine. The WebGetIfNotExist command will only download a file if it is not already found as FileOnDisk - particularly useful to download a zipped file once from an internet server and ignore the download the following times the script is executed.

- It is assumed that both machines (local and remote) are prepared for this operation.
- No checks are done to ensure that the local machine is online or that is has enough disk space to download the file.
- The remote server must also be accessible and the file must be available for download.
- Only direct download links are accepted, dynamic internet pages are not valid.

MD5 check is an additional parameter that can be added to ensure that your file matches a security check based on the MD5 hash calculation.

- This is an optional parameter, only needed for cases when it is important to only accept a file that passes on this download integrity test.
- If a file fails this test it won't be copied over to the final destination as FileOnDisk.

The optional parameters of **Ask** and **Timeout** are only used *if enabled*, and if enabled, will display a question dialog box, and wait for a response. This can be used to let the user skip over a download operation.

- To use this feature, the line "Ask before WebGet" in the Winbuilder Tab TOOLS\OPTIONS has to be checked.
- To use optional timeout, the line "Use Timeout when asking" in the Winbuilder Tab TOOLS\OPTIONS has to be checked.

If the target filename folder already exist, it will  ${f NOT}$  be created. See  ${f \underline{WebGet}}$  for alternative behavior)

### **Example:**

In this example we download the file iview423.zip from server foo.com to the folder c:\IrfanView\iview423.zip **ONLY** if the file does not already exist in that location, and

- only if MD5sum is fd490d07cc236cb44ef29abfd9daf13d
- with a question dialog's text string displaying 'ZIP-file'
- and automatically accept the dialog (i.e. timeout) after 10 seconds

WebGetIfnotExist,http://foo.com/iview423.zip,c:\IrfanView\iview423.z

# **Registry Handling**

These commands are written to write and read values from the local registry on your windows machine. You can also load the hives from the projects being built at any given time.

These registry operations are valid for any Windows Platform, you should however note that Vista (and later versions of windows) restricts the specific operation to load/unload hives - it is necessary to disable the UAC (User Account Control) before running projects that need this support.

#### List of available functions

- RegWrite
- RegRead
- RegDelete
- RegHiveLoad
- RegHiveUnload
- RegImport
- RegWriteBin
- RegReadBin
- RegMulti

# RegWrite

#### **Syntax**

RegWrite, <HKEY>, <KeyType>, <Section>, <Keyname>, <Value>

This command will create a new registry entry as specified.

• HKEY is the "Hive Key" where the key is to be placed. These hives are files that are loaded by your machine when booting and compose the registry structure. By default, most scripts use HKLM as the work hive.

#### Valid HKEY values

- HKLM
- HKCU
- HKCR
- HKEY USERS
- HKEY\_CURRENT\_CONFIG

Details on the functions of each hive can be found at the Microsoft Site.

• Keytype will specify how the data in the registry key should be interpreted.

#### Valid key type values

- 0x0 Empty key
- 0x1 String writes a text value
- 0x2 Expanded String will expand any variable value contained inside %%. (e.g. %temp%)
- 0x3 Binary data adds data in binary format. Each byte is specified by groups of two digits splitting each value with commas
- 0x4 Dword writes a numeric value
- 0x7 Writes multiple NULL separated strings
  - Note: See below for other types
- Section is the complete path to the section where the key is supposed to be

#### written

• Keyname and value are similar to the method for INI files, please ensure that the value you use is compatible with the key type.

## **Tips**

To create an empty section: Write a key using keytype as 0x0 and omit the entries for key and value - this will create a new section at the location you specify.

# **Example:**

 $RegWrite, HKLM, 0x4, Build \verb|\ControlSet001| Services \verb|\VgaSave| Device0, Defau| \\$ 

#### WB078 added:

- RegWrite gives a warning in the log, if the destination value already exists.
- If there is a parameter PRESERVE added, RegWrite does not overwrite an existing value. A warning is still given
- The warning can be suppressed by an additional parameterNOWARN
- The parameters PRESERVE, and NOWARN can be used simultaniously. The order does not matter.

#### Note:

This function can also take a decimal value for the registry type. This can be used to write newer types by translating the normal "hex" value to a decimal number. See this posting. As an example, there are double dword (aka "qword") registry values, designated type 0xb, as seen in the following exported value.

[HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\HotSta"LastTimeStamp"=hex(b): 00,00,00,00,00,00,00,00

Values like this can can be written to the registry using a decimal value of "11" instead of the equivalent hex value of "0xb", as shown in the following example.

RegHiveLoad, WB-Software, "%target\_sys%\config\software"
//transalate 0xb to integer 11
RegWriteBinEx, HKLM, 11, WB-Software, MyKey, 53, 00, 55, 00, 01, 08, a1, cc
RegHiveUnLoad, WB-Software

# RegRead

Reads a value from the registry.

RegRead, <HKEY>, <Section>, <Key>, <%Variable%>

• HKEY is the "Hive Key" where the key is to be placed. These hives are files that are loaded by your machine when booting and compose the registry structure. By default, most scripts use HKLM as the work hive.

#### Valid HKEY values

- HKLM
- HKCU
- HKCR
- HKEY USERS
- HKEY CURRENT CONFIG

Details on the functions of each hive can be found at the Microsoft Site.

• Keytype will specify how the data in the registry key should be interpreted.

#### Valid key type values

- 0x0 Empty key
- 0x1 String writes a text value
- 0x2 Expanded String will expand any variable value contained inside %%. (e.g. %temp%)
- 0x3 Binary data adds data in binary format. Each byte is specified by groups of two digits splitting each value with commas
- 0x4 Dword writes a numeric value
- 0x7 Multiple NULL separated strings
- Section is the complete path to the section where the key is supposed to be read.
- Keyname and value are similar to the method for INI files, please ensure that the value you use is compatible with the key type.

## **Example:**

 $RegRead, HKLM, Build \verb|\ControlSet001\\| Services \verb|\VgaSave|\\| Device0, Default Setup and Services \verb|\VgaSave|\\| Device0, Device0$ 

In this example the variable %myXresolution% will store the value of the mentioned key

# RegDelete

Deletes a key or value from the registry.

RegDelete, <HKEY>, <Key>[, Value]

# Example 1:

Delete a registry value

 $RegDelete, HKLM, Build \verb|\ControlSet001| Services \verb|\VgaSave| Device0, DefaultS| \\$ 

# Example 2:

Delete a key from the registry.

 $RegDelete, HKLM, Build \verb|\ControlSet001| Services \verb|\VgaSave| Device0|$ 

# ${\bf RegHive Load}$

Load an external registry hive onto your local registry system.

RegHiveLoad, <SectionName>, <HiveFilename>

#### Remarks

This command is based on Windows 32bit API and works under any Windows platform - under Vista/Windows7 it is necessary to disable the UAC (User Account Control) or Run as Administrator.

## **Example:**

In this example we will load the setupreg.hiv hive file from our work folder and place it under a registry section called Build

RegHiveLoad, Build, %targetdir%\I386\system32\setupreg.hiv

## RegHiveUnload

Unload an external registry hive from your local registry system that was previously loaded with the <a href="RegHiveLoad">RegHiveLoad</a> command or with another application capable of handling external hives.

RegHiveUnload, <SectionName>

#### Remarks

This command is based on Windows 32bit API and works under any Windows platform - under Vista/Windows 7 it is necessary to disable the UAC (User Account Control) or run as Administrator.

## **Example:**

RegHiveUnload, Build

## RegImport

This command will import an external registry file (\*.reg) **to your local registry system**.

RegImport, <RegFileToImport>

## Warning

- 1. As stated above, this adds information to the CURRENT systems registry. You should be careful and use this only with sections mounted using the proper commands to mount hives otherwise you might corrupt the current systems registry!
- 2. In addition, there is a known issue with binary data. See this posting on reboot.pro

## **Example:**

RegImport,c:\myFile.reg

## RegWriteBin / RegWriteBinEx

Write Wide Strings (e.g. Unicode) into the registry. The wide string is defined by a series of hexadecimal values, separated by comma.

RegWriteBin, <HKEY>, <KeyType>, <Section>, <Keyname>, <Value>

#### **Parameters**

1	HKEY	The "Hive Key" where the key is to be placed.
2	КеуТуре	Specify how the data should be interpreted
		0x1 - String - writes a text value
		0x2 - Expanded String - will expand any variable value contained inside %%. (e.g. %temp%)
		0x7 - Writes multiple NULL separated strings
3	Section	The complete path to the section where the key is supposed to be written
4	Key Name	The name of the name of one of the 'child keys' of a section. Value can be explicitly written list of hexadecimal values, separated by comma or a variable containing the list. Instead of the comma the variable must contain the escape #\$c
5	Value	Data to write

RegWriteBinEx has the same syntax, but writes every value of every type unchecked into the registry. It is in the responsibility of the script's author to make sure that only valid registry entries are created.

#### **Remarks**

The substrings of a 0x7 entry can be divided by ",00,00" bytes (without the quotes) or by the special character escape string #\$z.

#### **Examples:**

RegWriteBin using a comma seperated list of hex values

RegWriteBin, HKLM, 0x2, WB-Software\SOFTWARE\Classes\.bfc\ShellNew, Comm

RegWriteBin using a vairable containing a list of hex values

Set,%Value%,25#\$c00#\$c53#\$c00#\$c79#\$c00#\$c73#\$c00#\$c74#\$c00#\$c65#\$c0 RegWriteBin,HKLM,0x2,WB-Software\SOFTWARE\Classes\.bfc\ShellNew,Comm

# RegReadBin

Read a value from the key you specify.

RegRead, <HKEY>, <Section>, <Key>, <%Variable%>

#### This command will only retrieve values from the following keytypes:

- 0x1 String i.e. a text value
- 0x2 Expanded String will expand any system variable value contained inside %%. (e.g. %temp%)
- 0x7 Multiple NULL separated strings

**HKEY** is the "Hive Key" where the key is to be placed. These hives are files that are loaded by your machine when booting and compose the registry structure. By default, most scripts use HKLM as the work hive. Valid HKEY values are:

- HKLM
- HKCU
- HKCR
- HKEY USERS
- HKEY\_CURRENT\_CONFIG

**Section** is the complete path to the section where the key is to be read from

**Key** is the name of the 'child key' in that section to be read.

#### **Example:**

RegReadBin, HKLM, WB-Software\SOFTWARE\Classes\.bfc\ShellNew, Command, %

In this example the variable %Value% will store the value of the mentioned key

# RegMulti

Modify a multi-string registry entry.

RegMulti, <HKEY>, <Section>, <Key>, <Action>, [<Other Parameters>]

Based on the action specified, you may require one or more additional parameters

**HKEY** is the "Hive Key" where the key is to be placed. These hives are files that are loaded by your machine when booting and compose the registry structure. By default, most scripts use HKLM as the work hive. Valid HKEY values are:

- HKLM
- HKCU
- HKCR
- HKEY USERS
- HKEY CURRENT CONFIG

**Section** is the complete path to the section where the key is to be read from. **Note: The Section (i.e path to the Key) specified here** *MUST* **exist or an error will occur.**<sup>1</sup>

**Key** is the name of the 'child key' in that section to be read. **Note: The KEY specified here** *MUST* **exist or an error will occur.**<sup>1</sup>

**Action** may be one of the following keywords:

• APPEND - Writes a string at the end of the specified key.

RegMulti, <HKEY>, <Section>, <Key>, APPEND, <String>

- PREPEND Writes a string at the start of the specified key.
   RegMulti, <HKEY>, <Section>, <Key>, PREPEND, <String>
- BEFORE Writes a string before the matching sub-string.
   RegMulti, <HKEY>, <Section>, <Key>, BEFORE, <SubString>, <String>
- BEHIND Writes a string after the matchings sub-string.

RegMulti, <HKEY>, <Section>, <Key>, BEHIND, <SubString>, <String>

• PLACE - Writes a string at the specified index. If the string already exists in the current value, a warning is written and the current value is left unchanged.

```
RegMulti, <HKEY>, <Section>, <Key>, PLACE, <Index>, <String>
```

• DELETE - Removes the specified substring.

```
RegMulti, <HKEY>, <Section>, <Key>, DELETE, <SubString>
```

• INDEX - Queries the index of the specified string. If the substring does not exist, the returned value is 0.

```
RegMulti, <HKEY>, <Section>, <Key>, INDEX, <SubString>, %Index%
```

<sup>&</sup>lt;sup>1</sup> For more information on this and example, see <u>this post</u>.

## Retrieve

These commands will querry the OS and return variuos types of information about files and folders (or directories) The available sub-commands are:

- Retrieve, Dir Display a dialog box asking the user to select a folder
- Retrieve, File Display a dialog box asking the user to select a fil[ename
- Retrieve, File Size Calculate the size of a file
- Retrieve, File Version Retrieves version info from a specified file
- Retrieve, Folder Size Calculate the size of a folder
- Retrieve, MD5 Calculate the MD5 hash of a particular file

# Retrieve,Dir

Display a dialog box asking the user to select a folder.

Retrieve, Dir, < Directory Or Filename > , < % var % >

### **Parameters**

1	DirectorOrFileName	The initially selected folder
2		A user defined variable that will contain the full path of the selected directory

## **Example:**

You can use this example to open a dialog box asking the user to select a folder. This box will start in c:\. Once a file is selected, the %var% will contain the full path of this selected directory.

Retrieve, Dir, c:\,%var%

# Retrieve,File

Display a dialog box asking the user to select a file.

Retrieve,File,<Filename>,<%var%>

### **Parameters**

1	FileName	The initially selected folder and filter
2	%var%	A user defined variable that will contain the full path of the selected file

### **Example:**

This example will open a dialog box asking the user to select a file. This box will start in c:\ and will only display files that match the filter criteria which is \*.txt in this example. Once a file is selected, the %var% will contain the full filename of this selected file.

Retrieve, File, c:\\*.txt,%var%

# Retrieve,FileSize

Calculate the size of a file in bytes.

Retrieve,FileSize,<Filename>,<%var%>

### **Parameters**

1	Filename	The full path to the file
2	%var%	A user defined variable that will contain the size of the file

## **Syntax example:**

Retrieve, FileSize, c:\myFolder,%var%

# Retrieve, File Version

Retrieves version info from the specified file if it is available.

Retrieve,FILEVERSION,<Filename>,<%var%>

### **Parameters**

1	Filename	The full path to the file
2	%var%	A user defined variable used to store the version

## **Example:**

This example fills %var% with the file version string, as it can be seen by EXE-file's properties.

Retrieve,FILEVERSION,"c:\Program Files\IrfanView\i\_view32.exe",%var%

# Retrieve,FolderSize

Calculate the size of a folder in bytes.

## **Syntax**

Retrieve, FolderSize, <Directory>, <%var%>

### **Parameters**

1	Directory	The target directory
2	%var%	A user defined variable used to store the size of the folder.

# Retrieve, MD5

Calculate the MD5 hash of a file.

## **Syntax**

Retrieve, MD5, <Filename>, <%var%>

### **Parameters**

1	Filename	The full path of the file to hash
2	%var%	a user defined variable used to store the calculated hash.

#### **Remarks**

MD5 is a way to calculate a number known as a 'hash' or 'checksum' that is based on the content of a file.

It is most often used as a safety check to ensure that files are complete and unmodified when they are downloaded from a server.

This function will write the value of the MD5 result on %var%, you can compare against a security value written elsewhere to prove that your file is intact.

Retrieve, MD5, c:\MyText.txt,%var%

# **String Formatting**

WinBuilder has the following string formatting capabilities:

- BYTES
- CEIL/FLOOR
- CHARTOOEM / OEMTOCHAR
- DATE
- FILENAME / PATH / EXT
- INC / DEC / MULT / DIV
- LEFT / RIGHT
- LEN
- LTRIM / RTRIM / CTRIM / NTRIM
- POS
- REPLACE / REPLACEX
- SHORTPATH / LONGPATH
- SPLIT

# **StrFormat,BYTES**

### **Syntax**

StrFormat, bytes, <%VarResult%>

This command is used to convert a specific number of bytes from the given text to a more human readable format. Any amount of bytes specified will be converted to Kb, Mb, Gb and Tb.

StrFormat, bytes, 1440000, %myVar%

In this example %myVar% will contain "1.44Mb "

## StrFormat, [CEIL / FLOOR]

#### **Syntax**

StrFormat, <Action>, <%var%>, <%size%>

These functions will round up or down the variables value as specified to the nearest "size". Here "size" may be:

- number (e.g. 512)
- the character 'k' meaning 1024 (i.e. = kilobyte)
- the character 'm' meaning 1024 **1024 (i.e. = megabyte)**

The action selected is either:

- Ceil = round up
- Floor = round down

In this given example we are rounding up the value of %var% to the next multiple of %size%. The result is written back into %var% with the value of 40.

Set,%var%,37
Set,%size%,10
StrFormat,CEIL,%Var%,%size%

# StrFormat,[CHARTOOEM / OEMTOCHAR]

### **Syntax**

StrFormat, <Action>, <%string%>, <%VarResult%>

These functions convert between the ANSI charset and the DOS charset, and are useful when dealing with special characters like "umlauts" and other similar markings. Action can be either

- CharToOEM or
- OemToChar

## StrFormat, DATE

#### **Syntax**

StrFormat, Date, <%var%>, <yyyy-mmm-dd hh:nn am/pm>

This option is used to retrieve the current hour and date and output it to a variable using the format you prefer.

- yyyy year in 4 digit
- mmm 3 initial letters of the month
- dd day number
- hh hour number
- nn minute number
- am/pm indicates to use 12 hours cycle instead of 24 hours

The last parameter will define the way these values will be written to your variable. You can omit values that aren't needed and even add custom text inside.

In this example %myVar% will output the current year, month and day formatted as specified (eg. 2007-jun-07)

StrFormat, date, %myVar%, yyyy-mmm-dd

# StrFormat,[FILENAME / PATH / EXT]

#### **Syntax**

StrFormat, <Action>, <Filename>, <%VarResult%>

These Commands are used to extract specific information from the given text related to filenames.

List of possible Action values

- Filename returns the filename portion (no path)
- Path gives the path of the given text
- Ext outputs only the extension of the filename

Set, %DownloadURL1%, http://live.sysinternals.com/Bginfo.exe StrFormat, PATH, %DownloadURL1%, %var1% StrFormat, FILENAME, %DownloadURL1%, %var2% StrFormat, EXT, %DownloadURL1%, %var3%

#### which results in

%var1%=http://live.sysinternals.com/
%var2%=Bginfo.exe
%var3%=.exe

# StrFormat,[INC / DEC / MULT / DIV]

#### **Syntax**

StrFormat, <Action>, <%var%>, <n>

These commands are used to modify the value of a given number based on the arithmetic operation you select. It's also important to specify the value <n> to use to perform this operation.

• **Note: Only positive values allowed as the value of <n> -** in other words, you can't decrement by a negative value - use increment instead!

The result is written back to the variable (%var%).

• Note: While negative numbers are supported by this variable value, they can not be used as constant arguments to the command (as mentioned above).

List of possible Action values

- inc increase (%var% + n)
- dec decrease (%var% n)
- mult multiply (%var% \* n)
- div division (%var% / n)

where  $\mathbf{n}$  is the last parameter on this command and should always be a numeric value.

In this example we have a variable called %LoopAgain% with a value of 10. Once we increase the value of this value with the last parameter of this command we will get the same variable %LoopAgain% with a value of 15. This same example applies for all other available actions.

StrFormat, Inc, %LoopAgain%, 5

# StrFormat, [LEFT / RIGHT]

### **Syntax**

StrFormat, <Action>, <%string%>, <CharsToObtain>, <%var%>

This command gets a specified number of characters from the left or right end of a given variables content.

In this example we have a given string tomorrow and we want to get 3 chars from the left filled into %var1% with value tom, and 3 characters into %var2% (using StrFormat, RIGHT) with result row.

Set,%string%,tomorrow
StrFormat,LEFT,%string%,3,%var1%
StrFormat,Right,%string%,3,%var2%

# **StrFormat,LEN**

### **Syntax**

StrFormat, LEN, <%string%>, <%VarResult%>

Returns the number of characters in the string

there are 11 characters in the given string (which would be the value of %var1% StrFormat, LEN, "Hello World", %var1%

# StrFormat,[LTRIM / RTRIM / CTRIM / NTRIM]

#### **Syntax**

StrFormat, <Action>, <%string%>, <CharsToTrim>, <%var%>

These commands will trim off characters from the specified sting (or variable that contains a string) and out the result into the %var% specified. The action values are:

- LTrim trim from the left
- RTrim trim from the right
- CTrim trim a matching character
- NTrim trim any numbers from the **end** of the string

In this example we have a given string tomorrow and we trim 3 chars from the left. The remaining rest is filled into %var% with value orrow. The same syntax applies for StrFormat,RTRIM trimming 3 chars from the right resulting as tomor.

Set,%string%,tomorrow
StrFormat,LTRIM,%string%,3,%var%

CTRIM is used to trim the first and/or the last char of a given string, **BUT ONLY if the char to trim is matching with the first and/or the last char of the string.** If the char does not exist at the front or end of the string, it remains unchanged.

In this example we trim the backslash of given string Programs\ with the result Programs being written into %var%.

Set,%string%,Programs\
StrFormat,CTRIM,%string%,\,%var%

NTRIM is used to trim numbers from the end of a string. For example, it can be used to extract the class name from interface components. Here %var% will contain: pTextBox after this code is executed.

StrFormat, NTRIM, pTextBox12, %var%

# **StrFormat,POS**

### **Syntax**

StrFormat, POS, <String>, <SubString>, <%VarResult%>

find the position of the given SubString inside the specified String and return the result in %VarResult%

In this example we verify the position of (case insensitive) 'orl' in 'Hello World' with result 8.

StrFormat,POS,"Hello World",orl,%VarResult%

In this example we verify the position of lowercase 'l' in 'HELLO World' with result 10.

StrFormat,POSX,"HELLO World",1,%VarResult%

# StrFormat,[REPLACE / REPLACEX]

#### **Syntax**

StrFormat,REPLACE,<%string%>,<CharsToReplace>,<ReplacementChars>,
<%VarResult%>

Will "replace" a given number of characters inside a given string. It is **not** case sensitive.

• For a case sensitive version of this command, use (REPLACEX)

Set,%a%, BROWN
Set,%b%, yellow
Set,%source%,"The quick brown fox jumps over the lazy dog"
StrFormat,REPLACE,%source%,%a%,%b%,%VarResult%
StrFormat,REPLACEX,%source%,%a%,%b%,%VarResultSens%

In the first example we replace (case insensitive) brown by yellow resulting in new string being written to %VarResult%:

The quick yellow fox jumps over the lazy dog

In the second command, there is nothing replaced, because uppercase brown is not found, resulting the original string being placed into %VarResultSens%: - i.e.

The quick brown fox jumps over the lazy dog

# StrFormat,[SHORTPATH / LONGPATH]

#### **Syntax**

StrFormat, <Action>, <%string%>, <%VarResult%>

These commands will take a string and return either the "DOS 8.3" (i.e. ShortPath) path or the Longer, NT style path (LongPath).

# **StrFormat,SPLIT**

### **Syntax**

StrFormat, SPLIT, <%string%>, <DelimiterChar>, <Index>, <%VarResult%>

This command is used with the <u>Loop</u> command to split up a given string and act on the individual components.

(Index=0 --> number of items)

Set,%source%,"The quick brown fox jumps over the lazy dog" StrFormat,SPLIT,%source%,#\$s,0,%count% Loop,%ScriptFile%,Section,1,%count%

[Section]
StrFormat, SPLIT, %source%, #\$s, #c, %VarResult%

In this example we split the string "The quick brown fox jumps over the lazy dog" using the delimiter space (in it's escaped form #\$s) and get the single words in the separate loop passes. We use the "count" (returned as index=0) to drive the loop command.

This could also be used to get the nth delimited field in string read in from a file.

## **Text Files**

With this set of commands we can handle text files and modify them as needed. All these commands can be applied to either a single or multiple file using wildcards.

#### Description of commands

- TxtAddline
- <u>TxtReplace</u>
- <u>TxtDelLine</u>
- <u>TxtDelSpaces</u>
- <u>TxtDelEmptyLines</u>

### **TxtAddLine**

#### **Syntax**

TXTAddLine, <Filename>, <StringLine>, <Action>

With this command we can add a text line at a given location inside a text file, you can specify this location using an action option.

- Action options
  - **PrePend** Will add the text line on the top of the text file
  - **Append** The same as above but on the bottom of the file.
  - **Place** Will place the text line after a specified number of lines (see example for details)

With this syntax we add a line saying Hello World! in the 5th line counting from the top. If the text file doesn't have 5 lines, then it will be placed after the current last line. There is no limit to this count value.

TXTaddLine,C:\myFile.txt,Hello World!,place,5

# **TxtReplace**

## **Syntax**

TXTReplace, <Filename>, <OldString>, <NewString>

This command will look inside a given filename and replace all text that match the value of OldString and replace them with NewString

## **Example:**

All matches to Hello World! within a text file will be replaced by Have a good day

TXTReplace, C:\myFile.txt, "Hello World!", "Have a good day"

## **TxtDelLine**

#### **Syntax**

TXTDelLine, <Filename>, <StringToDelete>

This command will look for lines that match the StringToDelete value and will remove these lines from the text file.

StringToDelete doesn't need to be the value of the entire line, it only need to match the **beginning** of the line.

## **Example:**

All lines that start with "Hello" inside C:\myFile.txt will be removed.

TXTDelLine,C:\myFile.txt,Hello

# **TxtDelSpaces**

## **Syntax**

TXTDelSpaces, <Filename>

This is a very simple command that will delete all spaces inside a text file. It was created to make some specific INI files smaller in overall size.

## **Example:**

TXTdelSpaces,C:\myFile.txt

# **TxtDelEmptyLines**

## **Syntax**

TXTDelEmptyLines, <Filename>

This function will remove all empty lines from a given text file.

## **Example:**

TXTdelEmptyLines,C:\myFile.txt

# **Script Commands (Alphabetical)**

The available scripting commands are listed here in roughly alphabetical order.

Usually all command which compare etc. are case insensitive. But, there are some commands having a "X" at the end. These commands work like that ones without the "X", but case sensitive.

To access the list of all script commands in alphabetical order, see the <u>Script Commands (By Category)</u> page

# **AddInterface**

Allows interface components from another interface to be accessed locally in a script

#### **Syntax**

AddInterface, <Script>[, <Interface>], <prefix>

The optional <Interface> is used to read alternative interfaces, e.g. Interface-Language? The default value is to use the standard interface (Interface).

The prefix is used to distinguish to already existing interface values of the same name. The generated variables can be used as %prefix< original name</p>%. If prefix is omitted, also the underline in the name of the generated variables is omitted.

#### **Example:**

AddInterface, %ScriptFile%, Interface-German, DE

Values of the German interface are now accessible like

Echo, %DE\_pTextBox1%

if instead you used

AddInterface, %ScriptFile%, Interface-German,

Values of the German interface are now accessible like

Echo,%pTextBox1%

#### Note:

This command was added in Winbuilder 80; It will not work with older versions.

#### **ShellExecuteDelete**

#### **Syntax**

ShellExecuteDelete, <Action>, <Filename>[, Parameters][, StartDir], [%Exi

This function will launch a file using the settings you specify. WinBuilder will wait (pause) for the launched file or application to be closed before continuing to process the remaining commands. After it has been finished, WinBuilder will delete the application .

- **Action** is the specific method you wish to use to start the external file. If not specified, the default "Open" Will be used. Possible action values are:
  - Open will open the file using the default associated application. This
    means that if it is an executable, the program will be started, and
    possibly open a console window to display it's output.
  - **Hide** will also open the file, but it will be launched in hidden mode (i.e. any console output will not be viewable)
  - **Print** print a file using the default printer
  - **Explore** open an explorer window (used to display the contents of folders)
- **Filename** is the complete filename to where the file can be found, you can omit the dir part if this file can be found on the system paths.
- **Parameters** (optional) are the set of switches that you can pass along as command line parameters when executing the file.
- **StartDir** (optional) is the location that you wish to specify as your default work directory.
- **%ExitCode%** (optional) is the name of the variable that will be updated with the exit value returned by any executable started. This can be useful to validate the exit code value returned by the program. This variable must already exist, or a warning will be given, and WinBuilder will **not** be able to capture the return code.

## **Example:**

In this example we run a hidden executable, and delete it after it completes ShellExecuteDelete, hide, foo.exe

## **See Also**

• ShellExecute - has a similar behavior but does not delete the executable

#### **ShellExecuteEx**

#### **Syntax**

ShellExecuteEx, <Action>, <Filename>[, Parameters][, StartDir]

This function will launch a file using the settings you specify.

- **Action** is the specific method you wish to use to start the external file. If not specified, the default "Open" Will be used. Possible action values are:
  - **Open** will open the file using the default associated application
  - **Hide** will also open the file, but it will be launched in hidden mode (not user viewable)
  - **Print** print a file using the default printer
  - Explore open an explorer window (used to display the contents of folders)
- **Filename** is the complete filename to where the file can be found, you can omit the dir part if this file can be found on the system paths.
- **Parameters** (optional) are the set of switches that you can pass along as command line parameters when executing the file.
- **StartDir** (optional) is the location that you wish to specify as your default work directory.

The script will *NOT* wait (pause) for the launched file or application to be closed before continuing to process the remaining commands.

## **Example:**

In this example we can open an exporer window and pass in the name of the folder to open and do not wait for the executable to finish.

ShellExecuteEx,Open,explorer.exe,/e#\$c#\$q%OpenFolder%#\$q,%OpenFolder

#### **See Also**

• <u>ShellExecute</u> - has a similar behavior but will wait for the application to be finished **before** it will continue with the script processing.

# WinBuilder's Scripting "API"

The WinBuilder Scripting API (a.k.a. the "common" API) is a means to extend and add functionality to a WinBuilder project and to make application (and other) scripts more compatible across multiple projects. WinBuilder also provides a means to integrate these "macros" into the internal editor help system, and make them act and "feel" just like internal WinBuilder Commands

- How It works
- Hooking the API into your project
- The Common\_API.script file
- Defining the menu entries (syntax templates) for the magic wand

The current API project is still in a bit of "flux", however most commands have been around for some time and will continue to be supported in future versions.

A process to better manage the API is also under development.

#### How It works

The WinBuilder Scripting API is a set of macros, *nothing more nothing less*.

What makes it valuable is that the WinBuilder community has decided to use these same macro *definitions* across many projects. This allows individual scripts, especially for Applications, to be more easily used in different projects. If you know how to create a macro in a script, than you can easily figure out how the API is used. What makes this API process different than a script file containing ordinary macros is how it is "hooked into" the WinBuilder process.

Note: A WinBuilder projects does not **need** to use the API to be a valuable project. In fact, there are some projects that may not use ANY API, or may use the API process to implement their own API definition.

It is also important to note that the API just defines the Macro names and interface conventions. Individual projects may need to use a different internal implementation of the API, and that is perfectly OK - as long as the API follows the syntax and functionality defined.

# Hooking the API into your project

We will start by looking at what makes the API "turn on". If you look in **script.project** file you will see the following lines and Variables being defined. This first variable

```
//-- API File and Some Api project settings
%API%=%ProjectDir%\Build\Common_Api.script
```

tells WinBuilder what file contains the API, while this variable definition

```
%API_TYPE%=2
```

is used to tell the Common APIi that we are project type 2 (Vista/Win7). This is used internally in the API to make decisions about which implementation should be used when a command is called. Here are the currently defined type values

- 1 =Gena/LiveXP/NativeEx/PE1 based projects
- 2 =Vista/Win7 based projects
- 98=Windows98 based projects

The following line defines the name of the "section in common\_api file that contains the actual definitions of the API commands. In other words, what section has the code to be run when you use any given macro (e.g. Add\_shortcut etc.). We will look at that section of the Common API later.

```
%APIVAR%=ApiVar
```

There are some additional lines that should be added to the **script.project** file which help WinBuilder find and display items for the magic wand (i.e. WinBuilders internal help system). This is displayed in a pull-down menu, where the variable, as an example,

```
%APIDEF%=API
```

tells WinBuilder what to display for the top level API menu, and the variable

```
%APISUBDEF%=_Calculate, AddAutoRun, *AddPostConfig, *AddPreConfig, *AddT Add_Asso, *Add_Pin, Add_URL, Arch, Associate_File, Ask_License, CapiC
```

tells WinBuilder what the commands are in the API that have internal templates. **(This would be all on one line inside the script file)** In addition to defining the various variables, we need one last command to activate them at run time.

[Process]
AddVariables,%API%,ApiVar,GLOBAL

This command loads all commands in the common\_api.script from the [ApiVar] section we mentioned above as GLOBAL variables. We will get into this a bit more later, but know that this needs to happen in order for scripts to use the Common API commands.

## The Common\_API.script file

Ok now on to Common\_API.script. As mentioned above, we have defined several variables, and there is typically an [Api\_Def] section in the script file. For example:

```
[Api_Def]
%APIVAR%=ApiVar
%APIDEF%=API
%APISUBDEF%=_Calculate, AddAutoRun, *AddPostConfig, *AddPreConfig, *AddT
          Add_Asso, *Add_Pin, Add_URL, Arch, Associate_File, Ask_License, CapiC
          ExtractAndRunc, Others, *ReadENV, Registry, Require_File, *RequireSy
```

This section doesn't do anything! It is simply a reminder that you need to set these variables in your own script.project in order to use the Common API. **Editing this section will not do anything!** 

On the other hand, the <code>[!ApiVar]</code> Section is very important. This is the section we defined with the line <code>%APIVAR%=ApiVar</code> back in script.project and loaded into memory with the command

```
[Process]
AddVariables,%API%,ApiVar,GLOBAL
```

This is the part where the commands actually get mapped to their functions. i.e. we create the macro definitions. For example:

```
[ApiVar]
%ToRoot%=
%CapiArchApi%=
%CapiAdd_ShortcutProcess%=
%CapiSubFoldersToRoot%=Falsy
%CapiNoShortcutSubfolders%=
%CapiBSExplorerFix%=Nope
%CapiNowarnPreserveNodebugwarn%=
%reg_temp%=wb-hive
%reg%=%reg_temp%
%CopyMui%=Off
%WarnWhenSetApi%=False
%Shc_Mode%=0
_Calculate=Run,%API%,Calculate
Add_Asso=Run,%API%,Association%API_TYPE%
```

Add\_Shortcut=Run,%API%,AddShortcut%Capi\_Shortcut\_TYPE%%CapiAdd\_Short Add\_Pin=Run,%Api%,Add\_Pin\_Process%API\_TYPE% Add\_URL=Run,%API%,Add\_URL%API\_TYPE% AddAutoRun=Run,%API%,Add\_Auto\_Run%API\_TYPE% AddPostConfig=Run,%API%,Add\_Post\_Config%API\_TYPE% ----ETC ETC ETC------

An easy way to think of it is that whenever WinBuilder sees the command \_Calculate in any script it internally replaces the \_Calculate with

Run, %API%, \_Calculate

which will execute the [\_Calculate] section in %API% which we defined earlier in script.project as being %ProjectDir%\Build\Common\_Api.script

# Defining the menu entries (syntax templates) for the magic wand

The definitions are contained (as mentioned above) in the [API] Section. Note that line doesn't actually **do** anything - its just a reminder to developers that the next sections are menu definitions and not actually functions.

Following that line, you will see a bunch of sections all with names corresponding to the submenus we defined in the %APISUBDEF% variable in our project. this is where we actually define the commands to be displayed in the APIs magic wand menu. For example:

```
[API]
[ Calculate]
Full Syntax="//_Calculate, AutoIT Command(expression), ReturnVar(without)
Note="//Always use simple quote in expression ,use `#$c` in place of
MsgBox="_Calculate, MsgBox(2#$c'Test Calculate'#$c'Choice:'#$c5), Choi
InputBox=" Calculate,InputBox('Question'#$c 'Where were you born?'#$
[Add_URL]
Default Value="Add_URL, Favorites, <filename>, <webaddress>"
Full Syntax="Add_URL, [Favorites][Links]<TargetPath>, <filename>, <weba
Sample Favorites="Add_Url,,Google.url,http://www.google.com"
Sample Links="Add_Url,Links,Google.url,http://www.google.com"
Sample Favorites SubF="Add Url, Favorites\My Folder, Google.url, http:/
Sample AnyFolder="Add_Url,%target_win%,Google.url,http://www.google.
Sample Icon Url.dll="Add Url,, Google.url, http://www.google.com, UrlDL
Sample Icon Ico="Add_Url,,Google.url,#$psystemroot#$p\system32\blabl
[*Add_Pin]
Full Syntax="//Add_Pin, Type, Order(0, 1, 2, ..., 8, 9), path\)FileName, Titl
Default Value="Add_Pin,StartMenu,,%PE_Programs%\%ProgramFolder%\%Pro
Sample File="Add_Pin, Taskbar, , #$pSystemRoot#$p\calc.exe"
Sample Lnk="Add_Pin,StartMenu,0,"$Desktop\My Computer.lnk""
StartMenu="Add_Pin,StartMenu"
TaskBar="Add_Pin, Taskbar"
```

When winbuilder builds the api menu it looks at %APISUBDEF% and makes the top level entries based on the contents of this variable. Next it looks for sections inside the API script file that have the matching name of the top level entries and fills the top level menus with the lines in each section.

The sections are layed out like so:

- the name of command as shown in api menu = command to paste into the code editor when you select the command.
- -= makes a line/divider.

and that's the quick and dirty introduction to the common api. The rest of the code inside the script file are the functions that actually doing the work.

# NOTE: WinBuilder does not scan the sections. It only displays the contents of the section referenced by the corresponding name defined in %APISUBDEF%

- If it is not listed in any of the sections referenced by %APISUBDEF%, even if it has code to implament the functionality and is listed in the macro declaration, it won't have a Magic Wand entry.
- The variable %APISUBDEF% is only used by WinBuilder itself for building the API menu entries. The actual routines/functions are added in the [ApiVar] section.

# \_Calculate

Execute an AutoIT command that returns a value.

## **Syntax**

\_Calculate,AutoIT Command(expression),ReturnVar

## **Parameters**

1	AutoIt Command(expression)	AutoIt command to execute
2	ReturnVar	This is the name of the variable to receive the return value of the AutoIt command. It must be specified without the %

#### Remarks

This command uses a program called sCalculate to execute the AutoIt command. This program will be automatically downloaded to the winbuilder tools directory if it does not exist.

## **Tips**

- Always use simple quote in the command expression i.e. use [#\$c] in place of [,]
- Use the variable synatax (i.e. %ReturnVar% with the %) to access the return result in your code.
- See !AutoIT docs at <a href="http://www.autoitscript.com/autoit3/docs/">http://www.autoitscript.com/autoit3/docs/</a> for more information.

## **Example**

This will show a simple message box with Abort / Retry / Ignore buttons for 5 seconds

\_Calculate, MsgBox(2#\$c'Test Calculate'#\$c'Choice:'#\$c5), Choice

This will display an input box to ask the user to enter a string.

\_Calculate,InputBox('Question'#\$c 'Where were you born?'#\$c 'Planet

# This command has been superseded by **Associate\_File**

# Add\_Asso

This function is called to associate a file extension with an application.

## **Syntax**

Add\_Asso,Extension,[Type],[Title],[Filename],[Option],[UseRegAddBoot

## **Parameters**

1	Extension	File extension to be associated
2	Type	Set the type of the association. If not specified defaults to Open.
		Open - Launch the associated application
		Edit - Launch the associated editor and opens the document for editing
		Icon - Associate an icon with the extension
3	Title	Title to be associated with the extension
4	Filename	File to be associated. May be defaulted or may be a file name or full path. If it's not specified then it defaults to: %PE_Programs%\%programFolder%\%programExe% If only the file name is specified then the path is assumed to be: %PE_Programs%\%programFolder%
5	Option	This is optionally used to specify the DefaultIcon index. If the icon is contained with the program exe then this can be used with Open or Edit. Otherwise the Icon type can be used to specify an external DefaultIcon.
6	UseRegAddBoot	Defaults to False. If set to True then RegAddBoot is used to create the association.

## **Tips**

This function assumes the appropriate registry hive is loaded before the function is called. This is more efficient than <u>Associate\_File</u> if many associations are being created.

#### **Example**

This sets XnView.exe as the program to open files with extension Xnview.bmp using icon with index 2 from XnView.exe for the association. Title for the association is set to BMP Image.

```
[Process]
Hive_Load, HKU
Add_Asso, Ext, Open, BMP Image, %PE_Programs%\XnView\XnView.exe,, 2
Hive_Unload, HKU
```

If your scripts defines the variables %ProgramFolder% and %ProgramEXE% then the above example can be replaced with:

```
[Variables]
%ProgramEXE%=XnView.exe
%ProgramFolder%=XnView

[Process]
Hive_Load, HKU
Add_Asso, XnView.bmp,, BMP Image,, 2
Add_Asso, XnView.emf,, EMF Image,, 11
Add_Asso, XnView.gif,, GIF Image,, 3
Hive_Unload, HKU
```

which would generate this:

```
[Process]
Hive_Load, HKU
reg_add, 0x1, "%reg%\Classes\XnView.bmp", "", "BMP Image"
reg_add, 0x2, "%reg%\Classes\XnView.bmp\DefaultIcon", "", "%PE_Programs%
reg_add, 0x2, "%reg%\Classes\XnView.bmp\shell\open\command", "", "#$q%PE
reg_add, 0x1, "%reg%\Classes\XnView.emf", "", "EMF Image"
reg_add, 0x2, "%reg%\Classes\XnView.emf\DefaultIcon", "", "%PE_Programs%
reg_add, 0x2, "%reg%\Classes\XnView.emf\shell\open\command", "", "#$q%PE
reg_add, 0x1, "%reg%\Classes\XnView.gif", "", "GIF Image"
reg_add, 0x2, "%reg%\Classes\XnView.gif\DefaultIcon", "", "%PE_Programs%
reg_add, 0x2, "%reg%\Classes\XnView.gif\DefaultIcon", "", "%PE_Programs%
reg_add, 0x2, "%reg%\Classes\XnView.gif\Shell\open\command", "", "#$q%PE
Hive_Unload, HKU
```

# Add\_Pin (Win 7 Projects Only)

"Pin" a program to the Windows 7 taskbar or start menu.

## **Syntax**

Add\_Pin, Type, [Order], Path, [Title], [WorkFolder], [Parameters], [Icon], [

#### **Parameters**

1	Type	Defines the type of the pin i.e. where the pin appears.
		<b>StartMenu</b> = Pinned item will appear in the start menu.
		<b>TaskBar</b> = Pinned item will appear on the taskbar.
		<b>RecentPrograms</b> = Pinned item (which is a link already in the start menu or on the desktop) will be "pinned" to the Start Menu's "recent programs" area. <b>Note:</b> When Type = RecentPrograms, only parameter number three (i.e. Path) is used. Others are not applicable and should be left unspecified.
		***Added as a provisional part of the API Jan 2012.
2	Order	Specifies the order of the icons in the specified pinning area. If a script overwrites a previously defined Type with the same Order, than you will get a warning in log.  Note: Many Windows 7 projects use a separate script to define all the items to be pinned so this is not a problem in practice.
		<b>0-9</b> = This defines the initial order of the icons that are shown in the selected area. It can be changed at run time by drag and drop
		<b>Auto</b> = The first available place in the selected area is automaticaly found. This is the default action if the parameter is not specified
		This should either be left blank or the full path to the item to be pinned should be specified. This can either be the installed path of the file or the installed path of a shortcut file (.lnk file ) in the user profile. Logical names can be used to reference areas in the user profile i.e. \$Start_Menu, \$Quick_Launch, \$Desktop. If blank it defaults to %PE_Programs%\%ProgramFolder%\%ProgramEXE%.

3	Path	Examples: %SystemRoot%\calc.exe \$Start_Menu\Programs\Network\PENetwork.lnk \$Quick_Launch\PENetwork.lnk \$Desktop\My Computer.lnk  *Note: When Type = RecentPrograms, the path must refer toan existing link in the \$Start_Menu\Programs or \$Desktop areas.
4	Title	This specifies the title to associate with the shortcut created by the pinning process. If not specified then %ProgramTitle% is used.
5	WorkFolder	Optional parameter defining the current directory for the pinned program (default value = same Folder as the program set in parameter 3)
6	Parameters	Optional parameter defining any command line parameters for the pinned program.
7	Icon	Optional parameter defining which icon to use on the automatically created shortcut file. (No need to give path if IconFile is in same folder as the program set in parameter 3).
		<b>(Path\)IconFile</b> = File containing icon. Can be an ico, exe, or a dll file. First available icon in file will be associated with the shortcut fil
		(Path\)IconFile#\$cIndex = File containing icon. Can be an ico, exe, or a dll file. Icon with specified index will be associated with the shortcut file.  Note: must use excaped form of the "comma" as separator
		Optional parameter specifying how the window of the launched program should be shown. <b>Note:</b> the launched program can override this.
8	StartMode	You can also include an appended string indicating what hotkey is used to launch the program ( add the escaped form

		of the comma (#\$c) between the start mode and the hotkey ). Hotkeys are specified using the text form of the key seperated by + e.g. CTRL+ALT+A.
		<b>1</b> = Program is shown in normal mode. The default if the parameter is not specified
		2 = Program is shown minimized
		3 = Program is shown maximized
9	ToolTipText	Text to display when cursor is on the pinned icon of the program.

#### Limitations

- 1. Not all files can be pinned
- 2. Links (.lnk) pointing to .cpl files can not be pinned
- 3. Files located on removable devices (CD UFD etc.) can not be pinned.
- 4. The name of pin is automatically set based on the .lnk name or program name.
- 5. A maximum 10 items per type can be pinned. (i.e. 10 start menu and 10 taskbar items)

#### **Example**

```
Add_Pin, StartMenu
Add_Pin, Taskbar
Add_Pin, Taskbar, ,#$pSystemRoot#$p\calc.exe
Add_Pin, StartMenu, ,"$Desktop\My Computer.lnk"
Add_Pin, StartMenu, ,%PE_Programs%\%ProgramFolder%\MyTool_To_Pin.exe,"
Add_Pin, RecentPrograms, ,$Start_Menu\Programs\Accessories\Notepad.lnk
```

# **Add\_Shortcut**

Create shortcuts on the desktop, quicklaunch bar or start menu.

#### **Syntax**

# Parameters ( \* not supported by Vista / Win7 )

1	Type	Define the type of shortcut
		<b>Desktop</b> - Place an icon on the desktop
		<b>StartMenu</b> - Place an icon in the StartMenu\Programs (default value)
		QuickLaunch - Add an icon to the Quick Launch bar
		<b>SendTo*-</b> Add an icon on the right-click menu "Send To"
		<b>AutoStart or AutoRun -</b> Place an icon in the StartMenu\Programs\AutoStart
		<b>EditWith</b> - Add an icon on the right-click menu "Edit With %ProgramExe%"
		Accessories* - Place an icon in the StartMenu\Programs\Accessories
		AdministrativeTools* - Place an icon in the StartMenu\Programs\Administrative Tools
		SystemTools* - Place an icon in the StartMenu\Programs\Accessories\System Tools
		<b>StartMenuRoot</b> - Place an icon in the StartMenu Root
2	StartMenuFolder	Fill this value if you want to force the shortcut to be placed on a specific folder (let it empty for use the script foldername)
3	FullFileName	This switch is used if you want to create a shortcut different of %ProgramExe% (No need to give path if File is in same folder as %ProgramExe%) (default value = %Pe_Programs%\%ProgramFolder%\%ProgramExe%)
4	Title	Set your own title for this shortcut (default value = %ProgramTitle%)

5	Work Folder*	Decide in which folder should start the program (default value = same Folder as the program set in parameter 3)
6	Parameters	If you want to launch the program with command line parameters, write them here
7	IconFile	Specify which icon to use on the shortcut. (No need to give path if IconFile is in same folder as the program set in parameter 3)
		(Path\) IconFile - for icon type .ico or .exe
		(Path\) IconFile#\$cIndex* - for icon type .dll or .exe (simply add #\$c between IconFile and Index)
8	StartMode	Optional value to set how the window of the launched program should be.
		1 - Show
		2 - Show_Minimized (program will start minimized)
		<b>3</b> - Show_Maximized (program will start maximized)
9	ToolTipText*	Text to display when cursor is on the icon of the program

#### **Remarks**

- Add\_Shortcut is a generic command that will create a shortcut. On most projects, by leaving empty parameter 2, it will create this shortcut inside the Start Menu and will place it inside a subfolder that has the same name as the folder where the respective script is placed.
- You don't need to worry where the shortcut will be placed because each project will create them in the best possible way.
- This seems that it was coded exclusively to work with the explorer shell but it wasn't. If the user prefers (for example) to use xoblite, then the Add\_Shortcut will either use the equivalent shortcut or simply ignore the request if it is not possible to execute.

### **Tips**

If you want place an icon in  $StartMenu\Programs$  without subfolder use a point (.) as parameter 2

```
Add_Shortcut,StartMenu,.
Add_Shortcut,StartMenu,..
```

is same as

Add\_Shortcut,StartMenuRoot

#### **Example**

For a script called Test.script, located in folder Apps\CD\ProgTest\ like this:

```
[Variables]
%ProgramTitle%=Test
%ProgramEXE%=Prog.exe
%ProgramFolder%=AppTest

[Process]
...
you can add a shortcut called Test in StartMenu\Programs\CD\ProgTest for the executable Program Files\AppTest\Prog.exe by using:
Add_Shortcut

or
Add_Shortcut,StartMenu
To use icon Icon.ico from same folder as the %ProgramExe%, use
Add_Shortcut,StartMenu,,,,,,Icon.ico
or to use icon from I386\sytem32\shell32.dll with index 27, use
Add_Shortcut,Desktop,,,,,,#$pSystemRoot#$p\system32\shell32.dll#$c27
```

# Add

- Add, DesktopIni
- Add,ShortcutAdd,URL

# Add, Desktop Ini

Create a desktop.ini file in the selected folder allowing you to specify how a file system folder is viewed.

## **Syntax**

Add, DesktopIni, Type, [Folder], [Section], Value, Data

#### **Parameters**

1	Url	Selects common API function to execute. In this case its Add_DesktopIni. Its a mystery why Add_DesktopIni was not added to the common API.
2	Type	Specifies the target directory where the desktop.ini file will be created.
		<b>Desktop</b> - Desktop.ini will be created in the desktop folder.
		<b>StartMenu</b> - Desktop.ini will be created in the Start Menu folder.
		<b>StartMenuRoot</b> -Desktop.ini will be created in the root of the Start Menu folder.
		<b>Accessories</b> - Desktop.ini will be created in the Accessories folder.
		<b>QuickLaunch</b> - Desktop.ini will be created in the Quick Launch folder.
		<b>SendTo</b> - Desktop.ini will be created in the Send To folder.
		<b>Path</b> - Desktop.ini will be created in the path specified in the Folder parameter.
3	Folder	If Type is Path then this specifies the folder in which desktop.ini will be created.
4	!Section	Optional parameter that specifies the section in desktop.ini to be updated. If specified it must be. ShellClassInfo as that is the only section supported by windows.
5	Value	Value name to be updated in the selected desktop.ini file.
		<b>ConfirmFileOp</b> - Set this entry to 0 to avoid a "You Are Deleting a System Folder" warning when deleting or moving the folder.
		<b>NoSharing</b> - Not supported under Windows Vista or later. Set this entry to 1 to prevent the folder from being shared.

		<b>IconFile</b> - If you want to specify a custom icon for the folder, set this entry to the icon's file name. The .ico file name extension is preferred, but it is also possible to specify .bmp files, or .exe and .dll files that contain icons. If you use a relative path, the icon is available to people who view the folder over the network. You must also set the IconIndex entry.
		<b>IconIndex</b> - Set this entry to specify the index for a custom icon. If the file assigned to IconFile only contains a single icon, set IconIndex to 0.
		<b>InfoTip</b> - Set this entry to an informational text string. It is displayed as an infotip when the cursor hovers over the folder. If the user clicks the folder, the information text is displayed in the folder's information block, below the standard information.
6	Data	The value data.

#### **Remarks**

The Desktop.ini file is a text file that allows you to specify how a file system folder is viewed. The ShellClassInfo section, allows you to customize the folder's view by assigning values to the appropriate entries. File system folders are commonly displayed with a standard icon and set of properties, which specify, for instance, whether or not the folder is shared. You can customize the appearance and behavior of an individual folder by creating a Desktop.ini file for that folder. Will only work for Windows 7 if the desktop.ini file is the only file in the folder. This basically makes the feature useless as what is the point of having a custom icon for an empty folder.

#### **Example**

Add, DesktopIni, StartMenu,,, WordPad.lnk, shell32.dll#\$c-22069 Add, DesktopIni, Path, %Target\_PE%\AFolder,. ShellClassInfo, LocalizedRes Add, DesktopIni, Path, %Target\_PE%\AFolder,. ShellClassInfo, IconFile, #\$p Add, DesktopIni, Path, %Target\_PE%\AFolder,. ShellClassInfo, IconIndex, 25

# Add, Shortcut

See <u>Add\_Shortcut</u> for documentation, since they are just two different ways to call the same AP functionality

# Add, URL

Creates URL file links for default browser.

## **Syntax**

Add, Url, [Type], <FileName>, <WebAddress>, [IconFile], [IconIndex], [HotKe

#### **Parameters**

1	Url	Selects common API function to execute. In this case its Add_URL. Its a mystery why Add_URL was not added to the common API.
2	Type	Optional parameter specifying target directory where the URL file link will be created. Assuming favorites is chosen as the target directory then the links will automatically appear in the IE favorites menu. There is no current mechanism to create links that will appear in other browser's bookmarks menu.
		<b>Favorites</b> - URL link will be created in the IE favorites folder.
		Links - URL link will be created in the IE Links folder.
		<b>StartMenu</b> - URL link will be created in the StartMenu folder.
		<b>StartMenuRoot</b> - URL link will be created in the root of the Start Menu.
		<b>Desktop</b> - URL link will be created on the desktop.
		<b>Path</b> - URL link will be created in the path specified in the path part of the FileName parameter.
3	FileName	File name of the URL link file that will be created.
4	WebAddress	Web Address associated with the URL link.
5	IconFile	Optional parameter specifying full path of icon file to be associated with the URL link file. Can be specified as UrlDLL to reference the url.dll system file.
6	IconIndex	Optional parameter specifying the icon index from the IconFile. If not specified then the first icon in the specified file will be used.

7 HotKey	Optional parameter specifying the hotkey used to launch the URL by the default browser. !Hotkeys are specified using the
	text form of the key separated by + e.g. CTRL+ALT+A.

#### **Examples**

• Create in Favorites Folder:

```
Add, Url, , Google.url, http://www.google.com
Add, Url, Favorites, "My Folder\Google.url", http://www.google.com
Add, Url, , Google.url, http://www.google.com, UrlDLL
Add, Url, , Google.url, http://www.google.com, #$psystemroot#$p\syste
```

• Create in Links Folder

```
Add, Url, Links, Google.url, http://www.google.com
```

• Create in Start Menu

```
Add, Url, StartMenu, Google.url, http://www.google.com
```

• Create in Start Menu root folder:

```
Add, Url, StartMenuRoot, Google.url, http://www.google.com
```

• Create on Desktop:

```
Add, Url, Desktop, Google.url, http://www.google.com
```

• Create in specified folder:

```
Add, Url, Path, %target_win%\Google.url, http://www.google.com
```

# AddAutoRun

Configure a program to automatically run on startup.

## **Syntax**

AddAutoRun, [Title], [FileName], [Parameters], [StartMode]

#### **Parameters**

1	Title	Title to associate with the program to be automatically run at boot time. This defaults to %ProgramTitle%.
2	FileName	Filename and optional path of program to be run. This defaults to %PE_Programs%\%ProgramFolder%\%ProgramExe%.
3	Parameters	Command line parameters for the program being run.
4	StartMode	Used to specify if the program is run hidden and / or if the autorun program waits for the program to finish running.
		<b>1</b> = Wait for the program to finish running but show any window used by the program. This is the default start mode
		<b>2</b> = Don't wait for the program to run but show any window used by the program
		<b>3</b> =Run the program hidden and wait for the program to finish running
		<b>4</b> = Run the program hidden and don't wait for the program to finish running

#### **Remarks**

In PE1 projects the API calls AddShortcut, Autostart to create the run entry in the registry.

In PE2+ projects AddAutoRun will create an entry in the autorun section in the %PE\_CFG% configuration file.

This file is processed at boot time. The only difference between this and <a href="AddPostConfig">AddPostConfig</a> is the position in the startup sequence when the entry is run. This runs before the hardware detection is run and device drivers are installed.

In current PE2+ projects the %PE\_CFG% file is processed by:

• **nativeEx\_multiPE**: win7peldr.exe

• **VistaPE**: vpeldr.exe

• **Win7RescuePE**: win7peldr.exe

• Win7PE\_SE: shortcuts.exe

#### **Example**

This example shows how to automatically start a service at boot time. In this example the lanman service will be started. The loader program will wait for the service to start and hide the console window that otherwise would be shown.

AddAutoRun, "Start lanman service", net.exe, "start lanman", 3

# AddPostConfig (Vista / Win7 projects only)

Configure a program to automatically run at startup after drivers are installed.

## **Syntax**

AddPostConfig, [Title], [FileName], [Parameters], [StartMode]

#### **Parameters**

1	Title	Title to associate with the program to be automatically run at boot time. This defaults to %ProgramTitle%.
2	FileName	Filename and optional path of program to be run. This defaults to %PE_Programs%\%ProgramFolder%\%ProgramExe%.
3	Parameters	Command line parameters for the program being run.
4	StartMode	Used to specify if the program is run hidden and / or if the autorun program waits for the program to finish running.
		<b>1</b> = Wait for the program to finish running but show any window used by the program. This is the default start mode
		<b>2</b> = Don't wait for the program to run but show any window used by the progra
		<b>3</b> = Run the program hidden and wait for the program to finish running
		<b>4</b> = Run the program hidden and don't wait for the program to finish running

#### **Remarks**

This adds an entry to the postconfig section in the %PE\_CFG% configuration file. This file is processed at boot time by the vpeldr / win7peldr programs. The only difference between this and <a href="AddAutoRun">AddAutoRun</a> is the position in the startup sequence when the entry is run. This runs after the hardware detection is run and device drivers are installed.

#### **Example**

This example shows how to automatically start a service at boot time. In this example the lanman service will be started. The loader program will wait for the service to start and hide the console window that otherwise would be shown.

AddPostConfig, "Start lanman service", net.exe, "start lanman", 3

# AddPreConfig (Vista / Win7 projects only)

Configure a program to automatically run on startup before the Shell is started.

# **Syntax**

AddPreConfig, [Title], [FileName], [Parameters], [StartMode]

#### **Parameters**

1	Title	Title to associate with the program to be automatically run at boot time. This defaults to %ProgramTitle%.
2	FileName	Filename and optional path of program to be run. This defaults to %PE_Programs%\%ProgramFolder%\%ProgramExe%.
3	Parameters	Command line parameters for the program being run.
4	StartMode	Used to specify if the program is run hidden and / or if the autorun program waits for the program to finish running.
		<b>1</b> = Wait for the program to finish running but show any window used by the program. This is the default start mode
		<b>2</b> = Don't wait for the program to run but show any window used by the program
		<b>3</b> = Run the program hidden and wait for the program to finish running
		<b>4</b> = Run the program hidden and don't wait for the program to finish running

#### **Remarks**

This adds an entry to the preconfig section in the %PE\_CFG% configuration file. This file is processed at boot time by the vpeldr / win7peldr programs. The only difference between this, <a href="AddAutoRun">AddAutoRun</a>, or <a href="AddPostConfig">AddPostConfig</a> is the position in the startup sequence when the entry is run. This runs before the shell is started.

#### **Example**

This example shows how to automatically start a service at boot time. In this example the lanman service will be started. The loader program will wait for the service to start and hide the console window that otherwise would be shown.

AddPreConfig, "Start lanman service", net.exe, "start lanman", 3

# AddToPEShell (Vista / Win7 Projects only)

#### **Syntax**

AddToPeShell, Title, [Path\]FileName, [Default=True/False]

#### **Parameters**

1	Title	Title of shell program in PEShell shell program list. This defaults to %ProgramTitle%.
2	[Path\]FileName	File name and optional path of program to be run. This defaults to %ProgramFolder%\%ProgramExe%. If just the file name is supplied, the path will be set to %ProgramFolder%\FileName.
3	Default	Optional - Indicates if the title should be used as the default shell - PEShell will automatically select the default shell after the timeout period.

#### **Remarks**

This command is used to write information about a shell program to the configuration file (which is later processed, possibly at run time) so that the PE initialization process will run the right commands and start the right PE Shell (e.g. explorer etc).

This information is written to the [shells] section of the configuration file in a format similar to a shortcut - i.e. the title and path\filename separated by the "|" character.

If the optional parameter "Default" is passed in as "True", then the information about the program is also written as an "ini" style item in the [main] section of the config file, where the value of the title is set as the "defaultshell".

Currently this is only supported in Vista / Win7 projects.

# Example

This adds explorer.exe as the default shell to PEShell.

AddToPEShell, Explorer, explorer.exe, True

# Arch

Specify the source architecture supported by the script.

## **Syntax**

Arch, SourceType

#### **Parameters**

1	Source Type	Source Type supported by the script
		<b>x86</b> = Script only supports the x86 architecture
		x64 = Script only supports the x64 architecture
		x86 x64 = Script supports either the x86 or x64 architecture

#### Remarks

This function should be the first function called by a script in the process section.

#### **Tips**

If the source type is x86 then the only source architecture that is supported is x86. If the source type is x64 then, depending on if wow64 support is available, then the source architecture may be x86 or x64. If wow64 support is not available when using x64 source then only x64 source architecture will be supported.

For a project to support this command two project variables must be defined:

%SourceArch% - architecture type of the source

%TargetWOW64% - this by default is the same as %SourceArch%. If the project contains support for wow64 then it will be set by the wow64 script to x86.

If a script contains both x64 and x86 executables and / or dlls embedded in the script then its recommended that they be embedded in a folder named x86 or x64 so that a command like the following can be used:

unpack, %SourceArch%, , True, %target\_win%, system32, %ProgramEXE%

# Example

Script only supports the x86 architecture

Arch,x86

Script supports both x86 and x64 architectures

Arch, x86 | x64

# **Ask\_License**

Prompt the user to accept a license agreement before continuing.

# **Syntax**

 ${\tt Ask\_License,FullFilename,ID,SectionToRunBeforeExit}$ 

#### **Parameters**

1	FullFileName	Path to license file
2	ID	an identifier, defined by script developer to identify the license object. It must be unique for the script and should be simple, maybe something along the lines of 'Lic'.
3	SectionToRunBeforeExit	Section to run in %ScriptFile% (if License not found or not accepted) before exiting

#### **Remarks**

#### What happens:

- If the license file does not exist, the script is exited.
- If the license file exists, it is shown and the user is asked for agreement.
- If the user does not agree, then the script is exited.
- If the user agrees, then in %ScriptFile% a marker using the **ID** specified is set into the [Licenses] section, and script processing continues. See the example below.

On the next build, this marker is read and the license agreement window not shown again.

#### **Example**

This is how you show the license file:

```
[process]
// Unpack just the file LicenseAgreement.txt from section Folder to
Unpack,Folder,,LicenseAgreement.txt
Ask_License,%Target_Prog%\%ProgramFolder%\LicenseAgreement.txt,Lic,R

[Reject_Licence]
// If Licence not accepted delete %Target_Prog%\%ProgramFolder%
DirDelete,%Target_Prog%\%ProgramFolder%
```

After the script is first run and the license is accepted the following will be appended to the script so that the next time the script is run the license will not be shown. Obviously if you redistribute your script this section must be removed.

[Licenses] Lic=True

# Associate\_File

Associate a file extension with an application.

# **Syntax**

Associate\_File, Extension, [Type], [Filename], [Option], [UseRegAddBoot]

#### **Parameters**

1	Extension	File extension to be associated
2	Type	Set the type of the association. If not specified defaults to Open.
		<b>Open</b> = Launch the associated applicatio
		<b>Edit</b> = Launch the associated editor and opens the document for editin
		<b>Icon</b> = Associate an icon with the extensio
3	Filename	File to be associated. May be defaulted or may be a file name or full path. If it's not specified then it defaults to:  %PE_Programs%\%programFolder%\%programExe%  If only the file name is specified then the path is assumed to be:  %PE_Programs%\%programFolder%
4	Option	This is optionally used to specify the DefaultIcon index. If the icon is contained with the program exe then this can be used with Open or Edit. Otherwise the Icon type can be used to specify an external DefaultIcon.
5	UseRegAddBoot	Defaults to False. If set to True then <a href="RegAddBoot">RegAddBoot</a> is used to create the association.

# **Tips**

This function will mount and unmount the appropriate registry hive each time its called. If many associations are being created then its more efficient to use <a href="Add\_Asso">Add\_Asso</a> instead.

#### **Example**

This sets notepad.exe as the program to open files with extension ext using icon with index 153 from shell32.dll for the association.

```
Associate_File,Ext,Open,%SystemRoot%\system32\notepad.exe
Associate_File,Ext,Icon,%SystemRoot%\System32\Shell32.dll,-153
```

If your scripts defines the variables %ProgramFolder% and %ProgramEXE% then the above example can be replaced with:

[Variables]
%ProgramTitle%=Notepad
%ProgramEXE%=Notepad.exe
%ProgramFolder%=%SystemRoot%\System32

[Process]
Associate\_File,Ext

# CopyProgram

Copy all files from a specified directory to %ProgramFolder%

# **Syntax**

CopyProgram, SourceFolder, [TargetFolder], [Filename], [FileToDelete]

#### **Parameters**

1	SourceFolder	Source folder containing script files to copy. All files in this folder will be copied.
2	TargetFolder	Optional target folder to which the source folder will be copied. If it does not exist the folder will be created. It defaults to: %Target_Prog%\%ProgramFolder%
3	FileName	Used to test if filename exist in the source folder. If not a message will appear and the function will be skipped. It defaults to:  %ProgramExe%"
4	FileToDelete	Optional file name to delete in target folder

#### **Remarks**

It will usually be used when the script files are contained in a sub folder of the script folder or in the script folder itself. In the latter case you would use the FileToDelete parameter to delete the script file from the target folder.

#### **Example**

This will copy all files from a sub directory of the script directory named %ProgramFolder%. It will show an error if pcidevs.txt does not exist in this source directory.

CopyProgram,%ScriptDir%\%ProgramFolder%,,pcidevs.txt

# Hive\_Load

This command will load an external registry hive into your local registry system.

## **Syntax**

Hive\_Load, Hive

#### **Parameters**

Hive	Name of the registry hive to load
	<b>Default</b> - This is the current user hive ( HKCU ). Can also use <b>HKCU</b> or <b>User</b>
	<b>Software</b> - This is the software hive ( HKLM\Software ). Can also use <b>HKU</b> or <b>Machine</b>
	<b>System -</b> This is the system hive ( HKLM\System ). Can also use <b>HKLM</b>

## **Tips**

- The user should always place a <u>Hive\_Unload</u> call after all the other registry calls for this hive.
- The user should never reference CurrentControlSet. Always use ControlSet001.

This installs the snapman filter driver.

# **Hive\_Unload**

Unload an external registry hive that was previously loaded into your local registry system with <a href="https://linear.nload.">hive\_load</a>.

Hive\_Unload, Hive

#### **Parameters**

Hive	Name of the registry hive to unload	
	<b>Default</b> - This is the current user hive ( HKCU ). Can also use <b>HKCU</b> or <b>User</b>	
	<b>Software</b> - This is the software hive ( HKLM\Software ). Can also use <b>HKU</b> or <b>Machine</b>	
	<b>System -</b> This is the system hive ( HKLM\System ). Can also use <b>HKLM</b>	

This installs the snapman filter driver.

# Not\_Compatible [Removed as of CAPI 25rev61]

Generate an error message if the script is used in a none compatible project.

Not\_Compatible,OSType

#### **Parameters**

1	OSType	Project target type for which the current script is not compatible
	LiveXP	Script is not compatible with the LiveXP project
	VistaPE	Script is not compatible with the VistaPE project
	Win7PE	Script is not compatible with the Win7PE project

This will generate an error message at build time if the script is built in the LiveXP project.

Not\_Compatible,LiveXP

# Process\_Log

Log version info and current time a script is run into the project.info file

Process\_Log

Process\_Log

# **ReadENV**

Read all settings from project.ini into local script variables of the same name as the setting. Optionally you may specify setting to load from project.ini into a local script variable.

ReadEnv,[GlobalSetting],[LocalVariable]

#### **Parameters**

1	GlobalSetting	Optionally specifies the name of a setting in project.ini to load.
2	LocalVariable	Only used if the first parameter is specified and optionally allows the name of the local script variable into which the setting is to be loaded to be specified. By default the local variable name will be the same as the setting name.

The following loads all project settings, loads the Shell setting into a variable named %Shell%, loads the Shell setting into a variable named %ShellName%.

ReadENV, Shell ReadEnv, ShellName

# $Reg\_Add$

Add a value to the registry hive loaded by Hive\_Load.

Reg\_Add, ValueType,%reg%\KeyName,[ValueName],[Value],[PRESERVE],[NOWA

#### **Parameters**

1	ValueType	The type of the registry value
		0x0 - Registry key with no value
		0x1 - String Value
		<b>0x2</b> - Expanded String - will expand any variable value contained inside %%. (e.g. %temp%)
		<b>0x3</b> - Binary value - Each byte is specified by groups of two hex digits separated with commas e.g. 12,0F,25.
		<b>0x4</b> - DWORD integer value specified by a decimal digit number i.e. 23567.
		<b>0x7</b> - Writes null separated strings terminated with an extra null i.e. "ab","cd","efgh".
2	KeyName	Name of the registry key. Must be prefixed with %reg%\.
3	ValueName	Name of the registry value.
4	Value	Value. Format depends on the ValueType.
5	PRESERVE	If specified don't overwrite a value that already exists. A warning will be logged. Can be suppressed by specifying NOWARN.
6	NOWARN	Suppress the warning in the log file. The optional PRESERVE / NOWARN parameters can be specified in any order. One or both can be specified.

#### Remarks

• WB 77 RC2 or later: RegWrite gives a warning in the log, if the destination value already exists. The warning can be suppressed by specifying the NOWARN parameter.

## **Tips**

Creating an empty registry key: Write a key using keytype as 0x0 and omit the entries for ValueName and Value - this will create a new registry key at the location you specify.

# Reg\_Del

Delete a registry key or value from the registry hive loaded by Hive\_Load.

Reg\_Del, KeyName, [ValueName]

#### **Parameters**

1	KeyName	Key name to delete. Must be prefixed with %reg%\.
2	ValueName	Optional value name to delete. If not specified then the key and its contents will be deleted.

# Reg\_Import

Import a .reg file

Reg\_Import,RegFile

#### **Parameters**

1 RegFile Name of reg file to import.

This will download the latest version of reg2wbsprg.exe if not already downloaded and use it to convert the reg file to a script file using the -h -e -a parameters.

These cause the exe to add hive load / unload around the reg\_add commands, replace environment variables with variable names and use the common api.

It will then execute the Process section in the generated script to apply the registry commands to the appropriate registry hive.

As a script file is being generated and executed then obviously the reg file must only contain registry values of the types supported by winbuilder.

```
Reg_Import, temp.reg
  If the reg file contains the following:
  Windows Registry Editor Version 5.00
  [HKEY LOCAL MACHINE\SOFTWARE\NVIDIA Corporation\Global\Hybrid]
  "AddDeviceSequence"=dword:0000000
  [HKEY LOCAL MACHINE\SOFTWARE\NVIDIA Corporation\Global\Hybrid\Curr
  "Mode"=hex:03,00,00,00,00,00,00,00
  [HKEY_LOCAL_MACHINE\SOFTWARE\NVIDIA Corporation\Global\NvCplApi\Pc
  "ContextUIPolicy"=dword:00000002
  "CplGroupUIPolicy"=dword:00000002
  "TaskbarUIPolicy"=dword:00000002
  [HKEY LOCAL MACHINE\SOFTWARE\NVIDIA Corporation\Global\NvSvc]
  "DrvUpdCtrlCode"=dword:00000081
  then the following will be generated and imported into the software hive:
  [main]
  Title=temp
  Selected=False
  Level=10
  Author=PSC Program 'reg2WBS'
  Version=000
  Date=23-Aug-2009
  [Process]
  IniRead,%API%,Main,Version,%apiVersion%
  If, %apiVersion%, SMALLER, 14, Halt, "API version 14 or higher needed!"
  Hive Load, Machine
  reg_add,0x0,"%reg%\NVIDIA Corporation"
  reg_add,0x0,"%reg%\NVIDIA Corporation\Global"
  reg_add, 0x4, "%reg%\NVIDIA Corporation\Global\Hybrid", "AddDeviceSec
  reg_add, 0x3, "%reg%\NVIDIA Corporation\Global\Hybrid\Current", "Mode
  reg_add, 0x0, "%reg%\NVIDIA Corporation\Global\NvCplApi"
  reg_add, 0x4, "%reg%\NVIDIA Corporation\Global\NvCplApi\Policies", "C
  reg_add, 0x4, "%reg%\NVIDIA Corporation\Global\NvCplApi\Policies", "C
  reg_add, 0x4, "%reg%\NVIDIA Corporation\Global\NvCplApi\Policies", "I
  reg_add, 0x4, "%reg%\NVIDIA Corporation\Global\NvSvc", "DrvUpdCtrlCoc
  Hive_Unload, Machine
```

# RegAddBoot

Write to the registry at boot time.

RegAddBoot, Hive, ValueType, KeyName, ValueName, Value

#### **Parameters**

1	Hive	Hive to be updated at boot time
		<b>HKLM</b> = The HKEY_LOCAL_MACHINE hive
		<b>HKCU</b> = The HKEY_CURRENT_UER hive
2	ValueType	The type of the registry value
		0x0 = Registry key with no valu
		0x1 = String Valu
		<pre>0x2 = Expanded String - will expand any variable value contained inside %<var>%. (e.g. %temp%</var></pre>
		<b>0x3</b> = Binary value - Each byte is specified by groups of two hex digits separated with commas e.g. 12,0F,25
		<b>0x4</b> = DWORD integer value specified by a decimal digit number i.e. 23567
		<pre>0x7 = Writes null separated strings terminated with an extra null i.e. "ab","cd","efgh"</pre>
3	KeyName	Name of the registry key.
4	ValueName	Name of the registry value.
5	Value	Value. Format depends on the ValueType.

#### **Remarks**

Some programs don't correctly support expanding of environment variables in registry values. For those cases the registry updates can be delayed till PE boot time when the value of the variable is known. This allows the fully expanded value to be written to the registry instead of the %variable%.

# Register\_File ( LiveXP / NativeEx Projects Only )

Registers a COM or ActiveX object.

## **Syntax**

Register\_File,FileName,RegType

### **Parameters**

1	FileName	File to register
2	RegType	Type of registration to perform
		<b>Install</b> - Register the file using the DllInstall entry point ( /i on regsvr32 )
		<b>Register</b> - Register the file using the DllRegisterServer entry point

#### **Remarks**

This uses one of two methods to perform the registration:

- If %dllregmethod% project variable exists and is equal to Build-Time it will use the RegRedirect program to perform the registration. This program as its name suggests captures the registry updates and redirects them to the current software registry hive loaded by winbuilder.
- If %dllregmethod% project variable exists and is any other value or if the %dllregmethod% project variable does not exist it will write an entry in the software registry hive to perform a regsvr32 on the file at boot time. It will use an entry in Microsoft\Windows\CurrentVersion\RunOnceEx\550 to perform this registration.

#### **Tips**

- 1. If the file is to be registered at build then the file name should include path information to locate the file in the source or in the script directories.
- 2. If the file is to be registered at boot time then the file name should include run time path information or it should be locatable via the current path environment variable.

## Example

Register\_File,%SystemRoot%\system32\shell32.dll,Install

## Require\_File

This function is used to copy a source file from a source folder to a target folder expanding the file if its compressed. It will in addition copy any language resource files associated with the file.

## **Syntax**

Require\_File, SourceFile, [SourceFolder], [TargetFolder], [PRESERVE] | [Ov

### **Parameters**

1	SourceFile	Name of the file to copy. For XP based projects this should be the final expanded file name not the compressed file name.  This may optionally contain a path which will be relative to the SourceFolder / TargetFolder parameters.
2	SourceFolder	Folder containing the file to copy. This defaults to %Source_Sys%.
3	TargetFolder	Folder to which the file is to be copied. This defaults to %Target_Sys%.
4	PRESERVE	By default if the target file exists it will be overwritten - this can be explicitly requested by . This parameter is used to not copy the file if it already exists in the target folder. A warning will be generated in the log file if the target file exists. The optional NOWARN parameter can be used to suppress this warning.
5	NOWARN	By default a warning will be generated in the log file if the target file exists. This parameter is used to suppress the warning.
6	MUISource	For XP based projects only this parameter optionally allows the folder containing the language specific files associated with the source file to be specified.

#### **Remarks**

- The expansion of compressed files functionality is specific to XP based projects where many source files are compressed. These compressed files use the cabinet archiving method and usually have an identifiable \_ as the last character on their extensions. (e.g. myFile.Tx\_).
- Vista/Win7 source does not use explicitly compressed source files instead the source is contained within a WIM file which is compressed.

## **Tips**

• The PRESERVE parameter must be parameter 4. The NOWARN parameter must be parameter 5.

#### **Example**

This always copies shlwapi.dll, only copies swenum.sys if it does not exist in the target directory but logs a warning, only copies swenum.sys if it does not exist in the target directory and supresses the warning if it already exists in the target directory.

```
Require_File, shlwapi.dll
Require_File, shlwapi.dll, , , PRESERVE
Require_File, shlwapi.dll, , , PRESERVE, NOWARN
```

This always copies swenum.sys from the %Source\_Sys%\drivers folder to the %target\_sys%\drivers folder

Require\_File, drivers\swenum.sys

## Require\_FileQ

This function is used to copy or expand a single file to %target\_sys% based on app script architecture.

## **Syntax**

Require\_FileQ,SourceFile

## **Parameters**

1 SourceFile Name of the file to copy. This should be the final expanded file name not the compressed file name.

#### **Remarks**

This is a simplified and much faster to execute version of <u>Require\_File</u>. Wildcards are not supported.

- For 32 bit source the 32 bit version of the system file will always be copied. For 64 bit source the default will be to copy the 64 bit version of the system dll. If a script only supports 32 bit then copying of the 32bit system can be forced by setting the %CapiScriptArch% variable.
- The expansion of compressed files functionality is specific to XP based projects where many source files are compressed. These compressed files use the cabinet archiving method and usually have an identifiable \_ as the last character on their extensions. (e.g. myFile.Tx\_).
- Vista/Win7 source does not use explicitly compressed source files instead the source is contained within a WIM file which is compressed.

#### **Example**

This always copies shlwapi.dll.

Require\_FileQ, shlwapi.dll

This always copies the 32 bit clbcatq.dll regardless of the source architecture.

Set,%CapiScriptArch%,x86
Require\_FileQ,clbcatq.dll

If the source architecture is 64 bit and for some reason it is required to copy both versions of a file in a single script then this will copy both versions of clcatq.dll. If the source is 32 bit it will copy the same file twice.

Require\_FileQ, clbcatq.dll Set, %CapiScriptArch%, x86 Require\_FileQ, clbcatq.dll

# RequireSysFile (Vista / Win7 Projects only )

Copy a source file from the %SystemFiles% source folder to the %TargetDir% target folder.

## **Syntax**

RequireSysFile,SourceFile,[NOWARN],[PRESERVE]

### **Parameters**

1	SourceFile	Name of the file to copy. This may optionally contain a path which will be relative to the %SystemFiles% / %TargetDir% folders.
2	NOWARN	By default a warning will be generated in the log file if the target file exists. This parameter is used to suppress the warning.
3	By default if the target file exists it will be overwritten. parameter is used to not copy the file if it already exists target folder. A warning will be generated in the log file target file exists. The optional NOWARN parameter can used to suppress this warning.	

#### Remarks

- The copy will always be from the mounted install.wim.
- It will in addition copy any language resource files associated with the file.

## **Tips**

The NOWARN and PRESERVE parameters can be specified in any order.

## **NOTE:**

This function has been superceded by the Require\_File function which works for all common API projects.

### **Example**

This always copies shlwapi.dll, only copies swenum.sys if it does not exist in the target directory but logs a warning, only copies swenum.sys if it does not exist in the target directory and supresses the warning if it already exists in the target directory.

```
RequireSysFile, windows\system32\shlwapi.dll
RequireSysFile, windows\system32\shlwapi.dll, PRESERVE
RequireSysFile, windows\system32\shlwapi.dll, NOWARN, PRESERVE
```

This always copies swenum.sys from the %Source\_Sys%\drivers folder to the %target\_sys%\drivers folder

Require\_File, windows\system32\drivers\swenum.sys

# RunFromCD (LiveXP / NativeEx Projects Only )

Force the current script program to run directly from the boot media.

## **Syntax**

RunFromCD,[BuildType]

### **Parameters**

1	BuildType	Forces the programs from the current script to run directly from the boot media instead of from the ram drive
		<b>True</b> - Force the current script programs to run from the boot media at %ProgramFilesFolder%\%ProgramFolder%
		<b>FolderName</b> - Force the current script programs to run from the boot media at the specified folder

### Remarks

See also **RunFromRam** 

## Example

RunFromCD, True

## RunFromRam

Force the current script program to run directly from memory, either from a RAM drive or from a .wim file. This allows the program directory to be writable if your booting from a CD/DVD.

## **Syntax**

RunFromRam,[True]

## **Parameters**

1		Optional parameter used to override the project build model so that the programs for the current script are run from the boot ram drive. If this parameter is not specified the default project build model is used. In the vista / win7 versions of PE this forces the program to be in the boot.wim.
---	--	--

#### **Remarks**

In general there are 2 build models:

- Boot the windows directory from the boot media
- Boot the windows directory from a ram drive. This option gives better performance but uses much more ram.

See also **RunFromCD** 

## Example

RunFromRam, True

## ScriptInterface,Read

Read the value of an interface element from a different script than the currently executing one.

# WARNING! May not be correctly documented

## **Syntax**

ScriptInterface, Read, %VarResult%, Filename, ComponentName, [Position][,

#### **Parameters**

1	%VarResult%	Variable used to receive the value of the interface element
2	FileName	Relative path to script file containing interface which is being accessed
3	ComponentName	Name of interface element being accessed
4	Position	Optional value specifying the position in the element from which the data will be accessed. It you do not specify this option the "value" of the element will be read.
5	Interface	Optional value specifying the interface section inside the script. This option allows you to read elements from scripts having multiple interfaces.

#### **Example**

Read the value of the pTextBox1 textbox from the Hijackthis script to a variable named %Result%

ScriptInterface, Read, %Result%, %ProjectDir%\Apps\HijackThis!.script, p

Read the value of the pTest\_TextBox1 textbox from the Hijackthis script to a variable named %Result%. In this case the textbox is on an interface called Interface-2 and we want to read if the textbox is visible. We check by reading position 2.

ScriptInterface, Read, %Result%, %ProjectDir%\Apps\HijackThis!.script,p

## ScriptInterface,State

Show or hide a component in the interface of either the current script or a specified script.

# WARNING! May not be correctly documented

### **Syntax**

ScriptInterface, State, ComponentName[, State][, Filename][, RefreshInter

### **Parameters**

1	ComponentName	Name of interface component to show/hide
2	State	Optional parameter indicating if the interface component should be hidden (0) or shown (1). If the parameter is not specified then 1 (show) is assumed.
3	FileName	File name of the script containing the interface element to hide or show.
4	Optional parameter indicating if the interface shaped refreshed. If True then the interface is refreshed or not specified then the interface is not refreshed.	
5	Interface	Optional value specifying the interface section inside the script. This option allows you to read elements from scripts having multiple interfaces.

### Example

Show pTextFile1 in the current script.

ScriptInterface,State,pTextFile1,1,,True

# ScriptInterface,Write

Change an element in the script interface.

### WARNING MAY NOT BE CORRECTLY DOCUMENTED

### **Syntax**

ScriptInterface, Write, NewValue, Filename, ComponentName[, Position][, In

### **Parameters**

1	NewValue	New value for the component
2	FileName	Relative path to script file containing interface which is being accessed
3	ComponentName	Name of interface element being accessed
4	Position	Optional value specifying the position in the element to which the data will be written. If you do not specify this option the "value" will be written.
5	Interface	Optional value specifying the interface section inside the script. This option allows you to read elements from scripts having multiple interfaces.

#### **Example**

Write the value "Interface Write Test" to the pTextBox1 textbox interface element in the Hijackthis script.

ScriptInterface, Write, "Interface Write Test", %ProjectDir%\Apps\Hijac

Write the value "Interface Write Test" to the pTest\_TextBox1 textbox interface element in the Hijackthis script. In this case the textbox is on an interface called Interface-2 and we want to change the caption of the textbox located at position 1.

ScriptInterface, Write, "Interface Write Test", %ProjectDir%\Apps\Hijac

### Select\_Case

Run a statement depending on the value of an expression.

### **Syntax**

Select\_Case, Var, ListVar1, [ListVar2], [Separator]

### **Parameters**

1	Var	Variable to test
2	ListVar1	List of possible value that Var set in parameter 1 should be equal , separated by a " "
3	ListVar2	Second list of value separated by a " " (should have same number of variables than ListVar1 set in parameter 2)
4	Separator	Set this parameter to 1 if you want use a comma "," as separator rather the default separator " "

#### **Return Value**

Always return the variable %Case% as result

- If parameter 3 is empty %Case% return the position of Var in ListVar1 set in parameter 2 (0 if not found)
- If parameter 3 is not empty %Case% return one variable of ListVar2, corresponding of the position of Var in ListVar1 set in parameter 2 (" " empty if not found)

#### **Example**

#### One

```
[Process]
ReadEnv, SourceType
Select_Case, %Source_Type%, "XP|W2003|VistaCD|Win7CD", "1|1|2|2"
If, %Case%, Equal, , Message, "This script is no compatible with W2000 so
Else,Run,Section_%Case%
[Section_1]
//Code for LiveXP / NativeEx with XP/W2003 source
Echo, "%Source_Type%"
[Section_2]
//Code for Vista / Win7 with DVD source
Echo, "%Source_Type%"
Two
[Process]
ReadEnv, SourceType
Select_Case, %Source_Type%, "XP|W2003|VistaCD|Win7CD"
Run, Section_%Case%
Ï¿½
[Section_0]
//Code for LiveXP / NativeEx with W2000 source and Vista / Win7 with
Echo, "%Source_Type%"
Message, "This script is no compatible with W2000 source or WAIK sour
Exit, "Invalid Source"
[Section_1]
//Code for LiveXP / NativeEx with XP source
Echo, "%Source_Type%"
[Section_2]
//Code for LiveXP / NativeEx with W2003 source
Echo, "%Source_Type%"
[Section_3]
//Code for Vista with DVD source
Echo, "%Source_Type%"
[Section 4]
//Code for Win7 with DVD source
```

Echo, "%Source\_Type%"

#### Three

Select\_Case,%Os\_Type%,"x86|amd64|ia64","http://www.nirsoft.net/utils
If,Not,%Case%,Equal,,Unpack,%Case%,-URL

## **Unpack**

Extract files from inside your script, an archive, or from a URL to a program folder in your project.

### Syntax 1

Unpack, [Section], [Archive], [Option], [Target], [ProgramFolder], [Filena

### **Parameters**

1	Section	This is the section in %ScriptFile% which contain the encoded files to extract. (default value = Folder)
2	Archive	Use this if you want extract an archive (only archive supported by 7z.exe, even an achive in exe format)
3	Option	If Parameter 2 is
		<b>archive (zip,7z,rar)</b> - Set it to False if you don't want to delete the archive (default value = True)
		<b>SFX archive (exe)</b> - Set it to -y for silent extraction
		<b>empty</b> - If you want extract only one file from section rather all files set its name here
4	Target	First part to select where to extract files, generaly one of this four values :
		%TargetDir% - Root
		<b>%Target_Prog%</b> - \Program Files (default value = %Target_Prog%)
		<b>%Target_Win%</b> - \I386 or \Windows
		<b>%Target_Sys%</b> - \I386\System32 or \Windows\System32
5	ProgramFolder	Second part to select where to extract files (subfolder) (default value = %ProgramFolder%)
6	Filename	This is use for test if exist %ScriptDir%\ProgramFolder\ [Filename], if true exit Unpack and copy files from there by CopyProgram command. (default value = %ProgramEXE%)

### Syntax 2

Unpack,[Filename], Action, [Param3], [Param4], [Param5]

#### **Parameters**

1	Filename	Depending on Action : FullFilename of Script, Archive or Web Link (default value = %SrciptFile%)
2	Action	If parameter 2 not match with one of following word syntax 1 is used
		<b>All</b> - Extract All Files from Script set in parameter 1
		<b>-All</b> - Extract All Files but only If they Not Exist from Script set in parameter 1
		<b>One</b> - Extract only One File ( the file set in parameter 4) from Script set in parameter 1
		<b>Archive</b> - Extract an Archive (set in parameter 4) and delete Archive
		-Archive - Extract an Archive (set in parameter 4) but don't delete Archive
		<b>SFX</b> - Extract a SFX Archive (set in parameter 4), execute and delete the SFX Archive
		<b>UnZip</b> - Unzip an Archive file (from filename set in parameter 1) with 7x.exe
		<b>URL</b> - Download and Unzip Archive file (from link set in parameter 1) and delete Archive file
		<b>-URL</b> - Download and Unzip Archive file (from link set in parameter 1) but don't delete Archive file
3	Param3	If Action equal :
		<b>All</b> or <b>-All</b> or <b>One</b> or <b>Archive</b> or <b>-Archive</b> or <b>SFX</b> then use this parameter to set the section inside the Filename set in parameter 1 which contain the encoded files to extract. (default value = Folder)
		UnZip - Use this parameter to set the target where you want to

		UnZip Files from inside the Archive set in parameter 1 (default value = %Target_Prog%\%ProgramFolder%)
		<pre>URL - Use this parameter to set the target where you want Download and UnZip files from the Archive set in parameter 1 (default value = %Target_Prog%\%ProgramFolder%)</pre>
		<b>-URL</b> - Set here the target where you want UnZip Files (from Downloaded Archive saved in parameter 4) set in parameter 1 (default value = %Target_Prog%\%ProgramFolder%)
4	Param4	If Action equal:
		<b>All</b> or <b>-All</b> - Set here the target where you want Extract Files from Script set in parameter 1 (default value = %Target_Prog%\%ProgramFolder%)
		<b>One</b> - Set here the filename you want extract from section set in parameter 3 from Script file set in parameter 1 (no default value , must be set)
		<b>Archive</b> or <b>-Archive</b> - Set here the Archive filename you want extract from section set in parameter 3 from Script file set in parameter 1 (no default value, must be set)
		<b>SFX</b> - Set here the SFX Archive filename you want extract from section set in parameter 3 from Script file set in parameter 1 (no default value, must be set)
		UnZip or URL - Nothing for this Action
		<b>-URL</b> - Set here the target where you want store the downloaded archive (default value = %ProjectTemp%)
5	Param5	If Action equal:
		One or Archive or SFX or -Archive - Set here the target where you want Extract Files (default value = %Target_Prog%\%ProgramFolder%)

#### **Remarks**

The Unpack command will use the folder called "Folder" as default place to keep all files that you attached using the "Create script" tool

• All parameters are optional, so when you wrote :

Unpack

default value for empty parameters are used if they are needed, so this command would become :

Unpack, Folder, ,%Target\_Prog%, %ProgramFolder%, %ProgramEXE%

since the variable %Target\_Prog% (defined in Script.project) is the "Program Files" folder, and the variables %ProgramFolder% and %!ProgramEXE% should be defined in section Variables of your script.

Only when an archive(7z,zip,rar - not for SFX archive) is set in parameter 2, the first thing that Unpack do, is look in %ScriptDir%\%ProgramFolder% if the file given in parameter 6 (default value = %ProgramExe%) exist. If the file exist, all files in this folder are copied by CopyProgram rather extracted from %ScriptDir%.

### **Tips**

If you want Unpack files in %Target\_Sys% you can use a point (.) as parameter 5

```
Unpack,,,,%Target_Sys%,.
```

is same as

Unpack,,,,%Target\_Win%,System32

#### Example (Syntax 1)

Unpack all files from section Folder to Program Files\%ProgramFolder%
Unpack

Unpack Archive.7z from section Doc to Program Files\%ProgramFolder%\Help (unzip the archive and delete it after)

Unpack, Doc, Archive.7z, , ,%ProgramFolder%\Help

Unpack Archive.7z from section Doc to Program Files\%ProgramFolder%\Help (but don't unzip the archive)

Unpack, Doc, , Archive.7z, ,%ProgramFolder%\Help

Unpack an Archive SFX (prog.exe) from section Folder to Program Files\%ProgramFolder% (after prog.exe is extracted, it is run with the parameter -y and then deleted)

Unpack,,prog.exe,-y

Unpack one file(myfont.ttf) from section Fonts to I386\Fonts or Windows\Fonts

Unpack,Fonts,,myfont.ttf,%Target\_Win%,Fonts

==Example== (Syntax 2) Unpack all files from %ScriptFile% from section Folder to Program Files\%ProgramFolder%

Unpack,, All

Unpack Archive.7z from %ScriptDir%\other.script from section Doc to Program Files\%ProgramFolder%\Help (unzip the archive and delete it after)

Unpack,%ScriptDir%\other.script,Archive,Doc,Archive.7z,%Target\_Prog%

Unpack Archive.7z from %ScriptFile% from section Doc to Program Files\%ProgramFolder%\Help (but don't unzip the archive)

Unpack,,-Archive,Doc,Archive.7z,%Target\_Prog%\%ProgramFolder%\Help

Unpack an Archive SFX (prog.exe) from %ScriptFile% from section Folder to Program Files\%ProgramFolder% (after prog.exe is extracted, it is run with the parameter -y and then deleted)

Unpack,,SFX,,prog.exe

Unpack one file(myfont.ttf) from %ScriptFile% from section Fonts to I386\Fonts or Windows\Fonts

Unpack,,One,Fonts,myfont.ttf,%Target\_Win%\Fonts

UnZip an Archive from %ScriptDir%\Test.7z to Program Files\%ProgramFolder%

Unpack,%ScriptDir%\Test.7z,UnZip

Download an archive from http://www.mysite.com/test.7z to Program Files\%ProgramFolder% and delete test.7z after UnZip it

Unpack, http://www.mysite.com/test.7z, URL

Download an archive from http://www.mysite.com/test.7z to %GlobalTemplates%\Apps\%ProgramFolder% but only if not exist test.7z in %GlobalTemplates%\Apps\%ProgramFolder% and UnZip test.7z from %GlobalTemplates%\Apps\%ProgramFolder% to I386\System32 or Windows\System32 (test.7z is not deleted)

Unpack,http://www.mysite.com/test.7z,-URL,%Target\_Sys%,%GlobalTempla

### **Additional Syntax**

Any pages listed here contain temporary documentation for newly added or changed syntax constructs that appear in the WinBuilder Engine. They will be migrated to the proper pages at some point in the future.

### **Index of Files**

# Google Code Project: <u>winbuilder</u>Generated: '2012-08-12'

- <u>ScriptExamples</u>
- add
- addInterface
- add asso
- add\_pin
- add shortcut
- addautorun
- addpostconfig
- addpreconfig
- addtopeshell
- <u>addvariables</u>
- arch
- ask license
- associate\_file
- beep
- begin end
- calculate
- copyorexpand
- copyprogram
- <u>dircopy</u>
- <u>dirdelete</u>
- dirmake
- dirmove
- echo
- encode
- exit
- expand
- extractallfiles
- extractallfilesifnotexist
- extractandrun

- extractfile
- <u>filebyteextract</u>
- <u>filecopy</u>
- <u>filecreateblank</u>
- filedelete
- <u>filerename</u>
- getparam
- <u>halt</u>
- <a href="mailto:hive\_load">hive\_load</a>
- <u>hive\_unload</u>
- if
- <u>ifBlock</u>
- ifComparison
- ifExist
- <u>ifExistMacro</u>
- <u>ifExistRegMulti</u>
- <u>ifLicense</u>
- ifNot
- <u>ifOnline</u>
- ifPing
- ifQuestion
- <u>iniaddsection</u>
- <u>inidelete</u>
- <u>inideletesection</u>
- <u>inimerge</u>
- iniread
- <u>iniwrite</u>
- iniwritetextline
- <u>loop</u>
- message
- not\_compatible
- packparam
- process log
- <u>readenv</u>
- reg add
- reg del
- reg\_import
- regaddboot
- <u>regdelete</u>

- reghiveload
- reghiveunload
- regimport
- register file
- <u>regmulti</u>
- regread
- regreadbin
- regwrite
- regwritebin
- require\_file
- require\_fileq
- requiresysfile
- <u>retrieve</u>
- retrieveDir
- retrieveFile
- retrieveFileSize
- retrieveFileVersion
- retrieveFolderSize
- retrieveMD5
- run
- runfromcd
- runfromram
- scriptinterface\_read
- scriptinterface\_state
- <u>scriptinterface\_write</u>
- select case
- set
- shellexecute
- shellexecutedelete
- shellexecuteex
- strformat
- strformatBYTES
- strformatCEIL
- strformatCHARTOOEM
- strformatDATE
- strformatFILENAME
- strformatINC
- strformatLEFT
- strformatLEN

- strformatLTRIM
- strformatMID
- strformatPOS
- strformatREPLACE
- strformatSHORTPATH
- strformatSPLIT
- <u>system</u>
- <a href="mailto:systemComp80">systemComp80</a>
- <u>systemCursor</u>
- systemErrorOff
- systemFileRedirect
- systemGetEnv
- systemGetFreeDrive
- systemGetFreeSpace
- systemHasUAC
- systemIsAdmin
- systemIsTerminal
- systemLog
- systemOnBuildExit
- systemOnScriptExit
- systemReScanScripts
- systemRebuildVars
- systemRefreshInterface
- systemRegRedirect
- systemSaveLog
- <u>systemSplitParameters</u>
- txtaddline
- txtdelemptylines
- txtdelline
- txtdelspaces
- <u>txtreplace</u>
- <u>u2o Examples WB v082</u>
- unpack
- visible
- wait
- wbVersionHistory
- wb\_ini\_settings
- wbadditionalsyntax
- wbbugs

- wbcmdline
- wbcommonapi
- wbcompat
- wbfilestructure
- wbindex
- wbinterface gui
- wbknownissues
- wblicense
- wboverview
- wbprojects
- wbscriptAttachments
- wbscriptFile Operations
- wbscriptIF Clauses and Command Blocks
- wbscriptINI\_Manipulation
- wbscriptMiscellaneous\_Commands
- <u>wbscriptNetwork\_Tools</u>
- wbscriptRegistry\_Handling
- wbscriptText\_Files
- wbscriptapplication
- wbscriptcommand
- wbscriptcommandsByCategory
- wbscriptfag
- wbscriptlevels
- wbscriptlinks
- wbscripts
- wbscriptsyntax
- wbscriptsyntax80
- <u>wbscriptvariables</u>
- wbtools
- wbtrouble
- wbwelcome
- webget
- webgetifnotexist

moved to ifBlock.wiki this file can be deleted.

### **StrFormat,MID**

### **Syntax**

StrFormat, MID, <%string%>, <PositionOfFirstChar>, <CharsToObtain>, <%var This command pulss a given number of characters out of the "middle" of a string

### **Example:**

In this example we have a given string tomorrow and we want to get 4 chars beginning with the third char. The result is filled into %var% with value morr.

Set,%string%,tomorrow
StrFormat,MID,%string%,3,4,%var%

### u2o\_Examples\_WB\_v082

- Intro
- Registry Handling
  - RegRead
    - Complex Example 1: Read a Value, put Date on a textbox and refresh script interface
- Miscellaneous Commands
  - Run / Exec
    - Complex Example 1: Associate an Extension (or more) with a different icon
    - Implicit example 1: Run by button
- Network Tools
  - WebGet
    - Example 1: Download a Zip File

### **Intro**

We will try to show real and explicit examples, with the necessary code for each action and references to their use. By this way you'll understand the use of each function (separately), as well as have the ability to modify sentences for fit what you need when you are creating your own script.

# **Registry Handling**

### **RegRead**

References: RegRead

#### **Syntax:**

RegRead, <HKEY>, <Section>, <Key>, <%Variable%>

# Complex Example 1: Read a Value, put Date on a textbox and refresh script interface

This is an Event *Click Button*. When you click the button, call the section: **pButton1** 

```
[Interface]
pButton1="Copy Your current key",1,8,414,201,121,25,pButton1,0,Fal

[pButton1]
RegRead,"HKLM","SOFTWARE\Acronis\DiskDirector","enterprise","%temp
Set,%pTextBox2%,"%temp_reg_data%",Permanent
System,REFRESHINTERFACE
```

Note: The Key and Value must exist to get the content (Data) of the Value

References: <u>Graphical Interface</u> | <u>RegRead</u> | <u>Set</u> | <u>System -></u> RefreshInterface

Similar Example: Run by button

## **Miscellaneous Commands**

#### Run / Exec

References: Run / Exec

#### **Syntax:**

Run,%ScriptFile%,<Section>[,parameters]

# Complex Example 1: Associate an Extension (or more) with a different icon

Simplify the association of extensions, you can call so through a single *Run* line.

#### How to use:

```
Run, %scriptfile%, SectionAssocExt, <P#1>, <P#2>, <P#3>, <P#4>, <P#5>, <P#6>
```

```
//P#1 = EXTENSION
//P#2 = EXTENSION_LINK
//P#3 = EXTENSION_DESCRIPTION
//P#4 = ICON_CONTAINER
//P#5 = ICON_CONTAINER_INDEX
//P#6 = APP_FULL_PATH
//P#7 = APP_PARAMETERS
```

**Note**: All parameters are required.

#### The code sample:

```
[Interface]
pCheckBox5="JPG, JPE, JPEG", 1, 3, 30, 150, 100, 16, True
pCheckBox6=BMP, 1, 3, 30, 165, 100, 16, True

[Variables]
%ProgramTitle%=FastStone Image Viewer
%ProgramFolder%=FastStone Image Viewer
%ProgramEXE%=FSViewer.exe
```

MessageOne

```
[Interface]
pButton1=ButtonShowThisText, 1, 8, 189, 260, 25, 25, MessageOne, False, False
[MessageOne]
Message, "You message here...#$x With new line #$c and comma.", Inform
  References: Graphical Interface | Message | Syntax Rules and Escape
```

Characters

Similar Example: Read a value, put date on a textbox and refresh script interface

# **Network Tools**

#### WebGet

References: WebGet

#### **Example 1: Download a Zip File**

The best way to get small size scripts, is to download the application zip package from its website. And obviously it's good, add the variables and code necessary for update the script easily.

```
[Variables]
%ProgramTitle%=FastStone Image Viewer
%ProgramFolder%=FastStone Image Viewer
%ProgramEXE%=FSViewer.exe
%SMProgramDir%=%pTextBox1%
// Download Config
%DownloadFileBaseName%=FSViewer46
%DownloadFile%=%DownloadFileBaseName%.zip
%DownloadUrl%=http://www.faststonesoft.net/DN/%DownloadFile%
// path where program templates are stored
%ProgramTemplate%=%BaseDir%\Workbench\Common\FastStone_Image_Viewe
[Process]
Run, %ScriptFile%, Download
[Download]
Echo, "Trying download %ProgramTitle% (%DownloadFile%)..."
If, Not, ExistDir, %ProgramTemplate%, DirMake, %ProgramTemplate%
// Download
If, Not, ExistFile, %ProgramTemplate%\%DownloadFile%, WebGet, %Download
If, Not, ExistFile, %ProgramTemplate%\%DownloadFile%, Echo, "File not c
If, Not, ExistFile, %ProgramTemplate%\%DownloadFile%, Echo, "Perhaps the
// Unpack zip file
If, ExistFile, %ProgramTemplate%\%DownloadFile%, ShellExecute, Hide, %T
  References: Variables | Echo | Run / Exec | If Clauses and Exist... /
  NotExist... | WebGet | ShellExecute | Syntax Rules and Escape
  Characters
```

- WinBuilder
  - Overview and Getting Started
  - The WinBuilder Scripting Language
    - Syntax Rules and Escape Characters
    - File Structure
    - Script Levels
    - Variables
    - Graphical Interface
    - Link Files
  - WinBuilder Tools
- WinBuilder F.A.Q. and other Information
  - WinBuilder.ini Settings
  - Command line usage
  - Compatibility
  - Known Issues
  - <u>Troubleshootings</u>
  - Reporting Bugs
  - License
  - Version History
- WinBuilder Projects
- Application Scripts
- Script Examples & Templates
- Script Commands (By Category)
  - Attachments
    - ExtractFile
    - ExtractAndRun
    - <u>ExtractAllFiles</u>
    - ExtractAllFilesIfNotExist
    - Encode
  - File Operations
    - CopyOrExpand
    - DirCopy
    - DirDelete
    - DirMove
    - DirMake
    - Expand
    - FileCopy
    - FileDelete

- FileRename
- FileCreateBlank
- FileByteExtract
- IF Clauses and Command Blocks
  - If Clauses
  - Begin / End Block
  - Exist / NotExist
  - Comparison Operations (Equal / NotEqual / Smaller / Bigger)
  - Ping
  - Online
  - License
  - Not
  - Question
  - ExistMacro
  - ExistRegMulti
- INI Manipulation
  - <u>IniWrite</u>
  - IniRead
  - IniDelete
  - IniAddSection
  - <u>IniDeleteSection</u>
  - <u>IniWriteTextLine</u>
  - <u>IniMerge</u>
- Miscellaneous Commands
  - Addvariables
  - Beep
  - Echo
  - Exit
  - GetParam
  - Halt
  - Loop
  - Message
  - PackParam
  - Run / Exec
  - Set
  - ShellExecute
  - System
    - System, Comp80
    - System, Cursor

- System, ErrorOff
- System,FileRedirect
- <u>System,GetEnv</u>
- System,GetFreeDrive
- System,GetFreeSpace
- System, HasUAC
- System, Is Admin
- System,IsTerminal
- System,Log
- System, OnBuildExit
- <u>System,OnScriptExit</u>
- System, Rebuild Vars
- System, RefreshInterface
- System,RegRedirect
- System, ReScanScripts
- System, SaveLog
- System, SplitParameters
- Visible
- Wait
- Network Tools
  - WebGet
  - WebGetIfNotExist
- Registry Handling
  - RegWrite
  - RegRead
  - RegDelete
  - RegHiveLoad
  - RegHiveUnload
  - RegImport
  - RegWriteBin
  - <u>RegReadBin</u>
  - RegMulti
- Retrieve Functions
  - Retrieve,Dir
  - Retrieve, File
  - Retrieve,FileSize
  - Retrieve, File Version
  - Retrieve, Folder Size
  - Retrieve, MD5

- String Formating
  - BYTES
  - CEIL / FLOOR
  - CHARTOOEM / OEMTOCHAR
  - DATE
  - FILENAME / PATH / EXT
  - INC / DEC / MULT / DIV
  - <u>LEFT / RIGHT</u>
  - <u>LEN</u>
  - LTRIM / RTRIM / CTRIM / NTRIM
  - POS
  - REPLACE / REPLACEX
  - SHORTPATH / LONGPATH
  - SPLIT
- Text Files
  - TxtAddline
  - TxtReplace
  - <u>TxtDelLine</u>
  - <u>TxtDelSpaces</u>
  - <u>TxtDelEmptyLines</u>
- Script Commands (Alphabetical)
  - AddInterface
  - Addvariables
  - Beep
  - CopyOrExpand
  - DirCopy
  - <u>DirDelete</u>
  - DirMake
  - o DirMove
  - Echo
  - Encode
  - Exit
  - Expand
  - ExtractAllFiles
  - ExtractAllFilesIfNotExist
  - ExtractAndRun
  - ExtractFile
  - FileByteExtract
  - FileCopy

- FileCreateBlank
- FileDelete
- FileRename
- GetParam
- Halt
- If Clauses
  - Begin / End Block
  - Comparison Operations
  - Exist
  - ExistMacro
  - ExistRegMulti
  - License
  - Not
  - Online
  - Ping
  - Question
- IniAddSection
- <u>IniDelete</u>
- IniDeleteSection
- IniMerge
- <u>IniRead</u>
- <u>IniWriteTextLine</u>
- Loop
- Message
- PackParam
- RegDelete
- RegHiveLoad
- RegHiveUnload
- RegImport
- RegMulti
- RegRead
- RegReadBin
- RegWrite
- RegWriteBin
- Retrieve
  - Retrieve, Dir
  - Retrieve,File
  - Retrieve, File Size
  - Retrieve, File Version

- Retrieve, Folder Size
- Retrieve, MD5
- Run / Exec
- <u>Set</u>
- ShellExecute
- ShellExecuteDelete
- ShellExecuteEx
- String Formating
  - BYTES
  - CEIL / FLOOR
  - <u>CHARTOOEM / OEMTOCHAR</u>
  - DATE
  - FILENAME / PATH / EXT
  - INC / DEC / MULT / DIV
  - LEFT / RIGHT
  - LEN
  - <u>LTRIM / RTRIM / CTRIM / NTRIM</u>
  - POS
  - REPLACE / REPLACEX
  - SHORTPATH / LONGPATH
  - SPLIT
- System
  - System, Comp80
  - System, Cursor
  - System, ErrorOff
  - System, FileRedirect
  - System, GetEnv
  - System, GetFreeDrive
  - System,GetFreeSpace
  - System, HasUAC
  - System, Is Admin
  - System,IsTerminal
  - System,Log
  - System, OnBuildExit
  - System, OnScriptExit
  - System, Rebuild Vars
  - System,RefreshInterface
  - System,RegRedirect
  - System, ReScanScripts

- System, SaveLog
- System, SplitParameters
- TxtAddline
- <u>TxtDelEmptyLines</u>
- TxtDelLine
- <u>TxtDelSpaces</u>
- TxtReplace
- Visible
- Wait
- WebGet
- WebGetIfNotExist
- API Commands (Alphabetical)
  - <u>Calculate</u>
  - Add Asso
  - o Add Pin
  - Add Shortcut
  - Add, Desktop Ini
  - Add, Shortcut
  - Add, URL
  - AddAutoRun
  - AddPostConfig
  - AddPreConfig
  - AddToPEShell
  - Arch
  - Ask License
  - Associate File
  - CopyProgram
  - Hive Load
  - <u>Hive\_Unload</u>
  - Not\_Compatible
  - Process Log
  - ReadENV
  - Reg\_Add
  - Reg\_Del
  - Reg\_Import
  - RegAddBoot
  - Register File
  - Require File
  - Require FileQ

- RequireSysFile
- RunFromCD
- RunFromRam
- ScriptInterface,Read
- <u>ScriptInterface,State</u>
- ScriptInterface,Write
- <u>Select\_Case</u>
- <u>Unpack</u>
- Additional Syntax

# Script Syntax Changes that appear in version 080

- Registry Handling
  - / RegWriteBinEx
  - 0
  - 0

This page describes new functionalty introduced with WinBuilder 080. Older WinBuilder versions cannot work with this functionality.

# **Registry Handling**

These commands are written to write and read values from the local registry on your windows machine. You can also load the hives from the projects being built at any given time.

These registry operations are valid for any Windows Platform, you should however note that Vista restricts the specific operation to load/unload hives - it is necessary to disable the UAC (User Account Control) before running projects that need this support.

## RegWriteBin / RegWriteBinEx

RegWriteBin, <HKEY>, <KeyType>, <Section>, <Keyname>, <Value>

This command will create a new or overwrite an existing registry entry as specified. The command is used to write Wide Strings (e.g. Unicode) into the registry. The wide string is defined by a series of hexadecimal values, separated by comma.

RegWriteBinEx has the same syntax, but writes every value of every type unchecked into the registry. It is in the responsibility of the script's author to make sure that only valid registry entries are created.

HKEY is the "Hive Key" where the key is to be placed. These hives are files that are loaded by your machine when booting and compose the registry structure. By default, most scripts use HKLM as the work hive.

#### Valid HKEY values

- HKLM
- HKCU
- HKCR
- HKEY USERS
- HKEY CURRENT CONFIG

KeyType will specify how the data in the registry key should be interpreted. Valid key type values for RegWriteBin

- 0x1 String writes a text value
- 0x2 Expanded String will expand any variable value contained inside %%. (e.g. %temp%)
- 0x7 Writes multiple NULL separated strings

Section is the complete path to the section where the key is supposed to be written

Keyname is the name of the name of one of the 'child keys' of a section. Value can be

- an explicitelly written list of hexadecimal values, separated by comma
- a variable containing the list. Instead of the comma the variable must contain the escape #\$c

The substrings of a 0x7 entry can be divided by ",00,00" bytes (without the quotes) or by the escape #z.

#### **Examples:**

RegWriteBin, HKLM, 0x2, WB-Software\SOFTWARE\Classes\.bfc\ShellNew, Comm Set, %Value%, 25#\$c00#\$c53#\$c00#\$c79#\$c00#\$c73#\$c00#\$c74#\$c00#\$c65#\$c6 RegWriteBin, HKLM, <math>0x2, WB-Software\SOFTWARE\Classes\.bfc\ShellNew, Comm

## RegReadBin

RegRead, <HKEY>, <Section>, <Key>, <%Variable%>

This command will read a value from the key you specify.

It uses the same components as described above with the difference that it will copy the value of a key into a variable.

This command will only retrieve values from the following keytypes:

- 0x1 String
- 0x2 Expandable String
- 0x7 Multiple String

#### **Example:**

RegReadBin, HKLM, WB-Software\SOFTWARE\Classes\.bfc\ShellNew, Command, %

In this example the variable %Value% will store the value of the mentioned key

## RegMulti

These commands uses the same components as described above.

• Writes a string at the specified place. If the string already exists in the current value, a warning is written and the current value is left unchanged.

```
RegMulti, <HKEY>, <Section>, <Key>, APPEND, <String>
RegMulti, <HKEY>, <Section>, <Key>, PREPEND, <String>
RegMulti, <HKEY>, <Section>, <Key>, BEFORE, <SubString>, <String>
RegMulti, <HKEY>, <Section>, <Key>, BEHIND, <SubString>, <String>
RegMulti, <HKEY>, <Section>, <Key>, PLACE, <Index>, <String>
```

Removes the specified substring.

```
RegMulti, <HKEY>, <Section>, <Key>, DELETE, <SubString>
```

• Queries the index of the specified string. If the substring does not exist, the returned value is 0.

RegMulti, <HKEY>, <Section>, <Key>, INDEX, <SubString>, %Index%