

Version 2.64 - (C)1999- 2002 Jonathan Bennett

(Email: support@hiddensoft.com)

AutoIt Homepage

AutoIt is a simple tool that can simulate key presses, mouse movements and window commands (maximize, minimize, wait for, etc.) in order to automate any windows based task (or even windowed DOS tasks).

AutoIt was primarily designed to assist in automatically installing software that cannot be automatically installed by other means. This is most useful during a PC rollout where hundreds or thousands of client machines need to be automatically installed. However, AutoIt is not limited to software installation and can be used to automate most simple windows tasks.

When AutoIt is executed, it reads a specified script file. This script file allows AutoIt to perform a number of functions, these include:

- Executing programs (Windows applications, DOS commands, etc.)
- Sending keystrokes and mouse clicks (all characters, keyboard layout independent).
- Window functions (e.g. minimizing, hiding, restoring, waiting for, activating).

- Simple text clipboard functions.
- Simple registry functions.
- Simple string and variable functions.

Scripts can also be compiled into standalone executables using **Aut2Exe**.

Best of all, its small (self contained .exe, no DLL's, no registry entries) and **FREE**!

There is also an ActiveX control version of AutoIt called **AutoItX**. This is a stand alone control that can be used to great effect under other scripting languages such as WSH/VBScript.

Still needed, non-English versions of this file! (In any format, as long as you are willing to help maintain) - Check the non-English downloads at the AutoIt Homepage to see what languages have currently been done. Email me for the base HTML files.

Intended Use

This program is intended for performing simple automation tasks in Windows. It is **not** intended as the "be-all and end-all" of scripting. When performing complex tasks (registry manipulation, text file parsing, complex loops, etc.) I would recommend using AutoIt in conjunction with a fully-fledged scripting language (such as <u>WSH</u> or <u>Kixtart</u> - both free) and using AutoIt to "fill in the gaps". The current ethos of AutoIt is "If it can't be done on a single line - it's probably too complicated". This approach has worked well so far, and helped to keep the size of AutoIt down (it is a **tiny** executable!).

I personally use AutoIt for large PC rollouts. I use Kixtart/WSH as a "back-end" to copy files, run files, edit the registry and then use it to call AutoIt when I need to perform configurations that can only be usually achieved by manual input (software installation, configuring control panel, changing drive letters, etc.).

License

Author: Jonathan Bennett

WWW: http://www.hiddensoft.com/AutoIt/

Email: jbennett@hidden.demon.co.uk

END-USER LICENSE AGREEMENT FOR THIS SOFTWARE

Important - read carefully:

This End-User License Agreement ("EULA") is a legal agreement between you (either an individual or a single entity) and the mentioned author of this Software for the software product identified above, which includes computer software and may include associated media, printed materials, and "online" or electronic documentation ("SOFTWARE PRODUCT"). By installing, copying, or otherwise using the SOFTWARE PRODUCT, you agree to be bound by the terms of this EULA. If you do not agree to the terms of this EULA, do not install or use the SOFTWARE PRODUCT.

SOFTWARE PRODUCT LICENSE

The SOFTWARE PRODUCT is protected by copyright laws and international copyright treaties, as well as other intellectual property laws and treaties. The SOFTWARE PRODUCT is licensed, not sold. The definition of SOFTWARE PRODUCT includes any files generated by the SOFTWARE PRODUCT, such as compiled script files in the form of standalone executables.

1. GRANT OF LICENSE.

This EULA grants you the following rights:

Installation and Use. You may install and use an unlimited number of copies of the SOFTWARE PRODUCT.

Reproduction and Distribution. You may reproduce and distribute an unlimited number of copies of the SOFTWARE PRODUCT; provided that each copy shall be a true and complete copy, including all copyright and trademark notices, and shall be accompanied by a copy of this EULA. Copies of the SOFTWARE PRODUCT may be distributed as a standalone product or included with your own product.

Commercial Use. If the SOFTWARE PRODUCT is to be included in a solution or product that will be sold for profit, you are required to acknowledge in your product documentation and/or help files, that this SOFTWARE PRODUCT was used. You are further required to provide a link to the web homepage specified at the top of this EULA.

2. COPYRIGHT.

All title and copyrights in and to the SOFTWARE PRODUCT (including but not limited to any images, photographs, animations, video, audio, music, text, and "applets" incorporated into the SOFTWARE PRODUCT), the accompanying printed materials, and any copies of the SOFTWARE PRODUCT are owned by the Author of this Software. The SOFTWARE PRODUCT is protected by copyright laws and international treaty provisions. Therefore, you must treat the SOFTWARE PRODUCT like any other copyrighted material.

MISCELLANEOUS

If you acquired this product in the United Kingdom, this EULA is governed by the laws of the United Kingdom. If this product was acquired outside the United Kingdom, then local

law may apply. Should you have any questions concerning this EULA, or if you desire to contact the author of this Software for any reason, please contact him/her at the email address mentioned at the top of this EULA.

LIMITED WARRANTY

NO WARRANTIES.

The Author of this Software expressly disclaims any warranty for the SOFTWARE PRODUCT. The SOFTWARE PRODUCT and any related documentation is provided "as is" without warranty of any kind, either express or implied, including, without limitation, the implied warranties or merchantability, fitness for a particular purpose, or noninfringement. The entire risk arising out of use or performance of the SOFTWARE PRODUCT remains with you.

NO LIABILITY FOR DAMAGES.

In no event shall the author of this Software be liable for any damages whatsoever (including, without limitation, damages for loss of business profits, business interruption, loss of business information, or any other pecuniary loss) arising out of the use of or inability to use this product, even if the Author of this Software has been advised of the possibility of such damages.

Because some states/jurisdictions do not allow the exclusion or limitation of liability for consequential or incidental damages, the above limitation may not apply to you.

[END OF LICENSE]

What does the license mean?

Licenses are particularly hard to follow, so here are my answers to the most frequently asked license questions:

GENERAL

Can I distribute the base AutoIt executable (AutoIt.exe) or compiled scripts as part of my script/solution. **YES.**

Can I modify the AutoIt executable or compiled scripts (changing resources, and version information in order to pass off AutoIt as your own program). NO. You are only allowed to change the icon in compiled scripts, and this facility is provided by Aut2Exe.

If you require an alternative license agreement then email me and make me an offer :)

NON-COMMERCIAL USE

For non-commercial use, a "mention" that you are using AutoIt in your product/solution and a link to the AutoIt homepage would be **very much appreciated**.

COMMERCIAL USE

You are **required** to mention that AutoIt was used in your

product/solution. You are also required to provide a link to the AutoIt homepage.

Note: Commercial-Use implies that you are selling your solution/product for profit. Using AutoIt within an organisation (for example, internal scripted builds) is not classed as commercial-use.

Running AutoIt

AutoIt can be executed from the command line, or by double-clicking on its icon from the Windows Explorer.

>>> N.B. The ONLY file that is <u>necessary</u> for AutoIt to run is "AutoIt.exe", putting this file in the path on any PC will allow you to run AutoIt scripts on that PC using the command line syntax below. <<<

Command Line Syntax

AutoIt is executed from the command line in the following ways (make sure AutoIt.exe is in your path, or you run the commands from the directory that AutoIt is installed):

AutoIt.exe /reveal

AutoIt.exe <filename of script file>

AutoIt.exe /c "ScriptCmd, ScriptParam"

Reveal Mode

When in "reveal" mode, the AutoIt window will stay topmost. As you click on other windows to make them active, AutoIt will display their window titles, the mouse coordinates relative to that window, and any other text on the window that may help to distinguish it from other windows of the same title. As with MS ScriptIt, not all text on a window can be "seen" - experiment!

Information on the AutoIt reveal screen can be highlighted with the mouse and copied to the clipboard with "CTRL-C", this can then be pasted into another program / text editor.

Script Mode

AutoIt will execute the script filename given on the command line. The file can have any extension (if you use the installation routine, **.aut** will be automatically registered. Right clicking on a **.aut** file will enable you to run or edit the script).

Single Command Mode

AutoIt will execute the script contained in quotes as a single command. This is veyr useful for running AutoIt commands from other batch files or WSH/kixtart without having to provide a separate script file. For example:

AutoIt.exe /c "WinWaitActive, Untitled - Notepad"

Using Command Line Parameters

Parameters can be passed to the AutoIt script from the command line when using the following syntax:

AutoIt.exe <filename of script file> param1 param2 param3......param_nn

(Note: previous versions of AutoIt required /**p** to be used to indicate that command line parameters were present, this is no longer required or supported).

These parameters are usable from the script file as the <u>ENV variables</u> %1%, %2%, %3%, etc. The variable %0% will also contain the number of parameters passed.

If a parameter needs to contain spaces, then enclose it in double quotes, e.g.:

AutoIt.exe testscript.aut "D:\A long pathname" var2

%1% would equal **D:\A long pathname**

%2% would equal **var2**

Window Titles and Window Text

A number of script commands have "Window Title" and "Window Text" as parameters.

"Window Title" is text that appears in the windows title bar.

"**Window Text**" is any other text on that window, e.g. text on buttons such as "OK" and "Cancel".

Having "Window Text" allows you to distinguish between multiple windows that have the same title.

Neither the title nor text needs to be completely specified. *i.e.* "**Untitled - No**" will match a window with the title "**Untitled - Notepad**".

See the text on the SetTitleMatchMode command for additions to this.

Use the <u>"/reveal"</u> mode to see what text is present on a window.

N.B. Windows titles and text are CASE SENSITIVE!!!!!

Conventions

Parameters are shown in angled brackets, optional parameters are shown in square brackets.

```
e.g.
<cmd>, <parameter1> [,<parameter2>]
```

This is a command, followed by a comma **and** parameter1, optionally followed by a comma **and** parameter2.

```
i.e.
<cmd>, <parameter1>
```

OR

```
<cmd>, <parameter1>, <parameter2>
```

Script File Syntax

The script file is a standard text file (created with any text editor, such as notepad.exe) that can be given any extension (although .aut is the preferred extension). Each line can contain either a command, a comment, a label, or be left blank.

N.B. The maximum number of characters per line is <u>16384</u>. Any extra characters will be ignored.

```
; Start of script
; Comments are indicated by a semi-colon

<Cmd>, <Cmd Parameters>
<Cmd2>, <Cmd2 Parameters>
<Cmd3>, <Cmd3 Parameters>

; This is a label, it must end with a colon (:) and it is used for commands such as Gosub and Goto.
label:

; End of script
Exit

; Start of the ADLIB section
[ADLIB]
```

<Title>, <Text>, <Cmd>, <Cmd Parameters>

The [ADLIB] Section

The [ADLIB] section is optional, and if present should be placed at the end of the script file. The ADLIB section is designed to cope with unexpected conditions. The ADLIB section consists of single line commands that check if a window is active and if so, send keys to it. The ADLIB section must start with the line "[ADLIB]".

The format of the a line in the ADLIB section is:

<Window Title>,<Window Text>,<Command>

e.g.
Run,Notepad.exe
WinWaitActive,Untitled - Notepad
Send,12345678#!{F4}
WinWaitClose,Untitled - Notepad

[ADLIB]

Notepad, save the changes?, Send, !n

Without the ADLIB section, this script would never end as Notepad would never

close until someone responded to the message "Do you want to save the changes?".

ADLIB commands must contain a window title and some text on the window. If there is no text on the window, or if you don't want to specify any, then you must use two commas.

e.g. Run,Notepad.exe

WinWaitActive,Untitled - Notepad Send,12345678#!{F4} WinWaitClose,Untitled - Notepad

[ADLIB] Notepad,, Send, !n

The "#include" Command

In an AutoIt script, other scripts can be included using the "#include" command:

#include <filename to include>

e.g.

include.aut

MsgBox, 0, Example, This is from the "include.aut" file Exit

script.aut

#include include.aut MsgBox, 0, Example, This is from the "script.aut" file Exit

Running the **script.aut** script will output the message box "This is from the 'include.aut' file", followed by "This is from the 'script.aut' file".

#include can be used anywhere in the main body of the script (not the ADLIB section) and will include the specified file at that position. (If the include file contains an ADLIB section, it will be added to the overall ADLIB section!).

Send Command Syntax

The "Send" command syntax is similar to that of ScriptIt and the Visual Basic "SendKeys" command. Characters are sent as written with the exception of the following characters:

"

This tells AutoIt to send an ALT keystroke, therefore "Send,This is text!a" would send the keys "This is text" and then press "ALT+a".

N.B. Some programs are very choosy about capital letters and ALT keys, i.e. "!A" is different to "!a". The first says ALT+SHIFT+A, the second is ALT+a. If in doubt, use lowercase!

'+'

This tells AutoIt to send a SHIFT keystroke, therefore "Send,Hell+o" would send the text "HellO". "Send,!+a" would send "ALT+SHIFT+a".

1 / 1

This tells AutoIt to send a CONTROL keystroke, therefore "Send,^!a" would send "CTRL+ALT+a".

N.B. Some programs are very choosy about capital letters and CTRL keys, i.e. "^A" is different to "^a". The first says CTRL+SHIFT+A, the second is CTRL+a. If in doubt, use lowercase!

'#'

The hash is used as a key delimiter to make a line easier to read. i.e. "Send, H#e#l#l#o" is the same as "Send, Hello".

Certain special keys can be sent and should be enclosed in braces:

N.B. Windows does not allow the simulation of the "CTRL-ALT-DEL" combination!

| Send Command | Resulting Keypress |
|---------------------|--------------------|
| {!} | ! |
| {#} | # |
| {+} | + |
| {^} | ^ |
| {{} | { |
| {}} | } |
| {SPACE} | SPACE |
| {ENTER} | ENTER |
| {ALT} | ALT |
| {BACKSPACE} or {BS} | BACKSPACE |
| {DELETE} or {DEL} | DELETE |
| {UP} | Cursor up |
| {DOWN} | Cursor down |
| {LEFT} | Cursor left |
| {RIGHT} | Cursor right |
| {HOME} | НОМЕ |
| {END} | END |
| {ESCAPE} or {ESC} | ESCAPE |
| {INSERT} or {INS} | INS |
| {PGUP} | PGUP |
| {PGDN} | PGDN |
| {F1} - {F12} | Function keys |
| {TAB} | TAB |

| {PRINTSCREEN} | PRINTSCR |
|------------------------|---|
| {LWIN} | Left Windows key |
| {RWIN} | Right Windows key |
| {NUMLOCK} | NUMLOCK |
| {CTRLBREAK} | Ctrl+break |
| {PAUSE} | PAUSE |
| {CAPSLOCK} | CAPSLOCK |
| {NUMPAD0} - {NUMPAD 9} | Numpad digits |
| {NUMPADMULT} | Numpad Multiply |
| {NUMPADADD} | Numpad Add |
| {NUMPADSUB} | Numpad Subtract |
| {NUMPADDIV} | Numpad Divide |
| {NUMPADDOT} | Numpad period |
| {APPSKEY} | Windows App key |
| {ALTDOWN} | Holds the ALT key down until {ALTUP} is sent |
| {SHIFTDOWN} | Holds the SHIFT key down until {SHIFTUP} is sent |
| {CTRLDOWN} | Holds the CTRL key down until {CTRLUP} is sent |
| {LWINDOWN} | Holds the left Windows key down until {LWINUP} is sent |
| {RWINDOWN} | Holds the right Windows key down until {RWINUP} is sent |
| {ASC nnnn} | Send the ALT+nnnn key combination |

To send the ASCII value A (same as pressing ALT+65 on the numeric keypad)

Send, {ASC 65}

Single keys can also be repeated, e.g.

Send, {DEL 4} Presses the DEL key 4 times

Send, {S 30} Sends 30 'S' characters

Send, +{TAB 4} Presses SHIFT+TAB 4 times

Special Characters

The slash'\', comma ',' and percent '%' are treated as special characters, and you must be careful when using them.

',' is used to separate parameters in the script file.

If you wish to use any of this characters literally, you must precede them with the slash '\' character, i.e. "\," or "\%". If you wish to use a literal '\' you must precede this also, i.e. "\\".

The "\n" and "\t" sequences can be used to denote new lines and tabs in the AutoIt output functions MsgBox and InputBox.

^{&#}x27;%' is used to indicate DOS variables.

Environment Variables

You can use DOS environment variables anywhere in an AutoIt script. As in DOS batch files, just include the variable enclosed in percent (%) symbols.

For example, if you want to send the current DOS path to the notepad window you could do:

Run, notepad.exe WinWaitActive, Untitled - Notepad Send, This is the DOS path %PATH%

e.g. This example sets the variable **test** and then outputs it in a messagebox.

SetEnv, test, This is some output MsgBox, 0, Example, %test%

N.B. Because of its special status, if you want to actually use a percent sign without it being changed into a variable, you must use the escape character '\' first.

e.g.

Send, This will send a single percent sign \%

You can also use "SetEnv" to set variables..

N.B. Many AutoIt commands specifically state that they take a variable as a parameter, in these cases the % symbol is NOT used. A number of the examples that come with AutoIt demonstrate this.

The Clipboard Variable

The variable **%CLIPBOARD**% is special, and when used will be substituted for the text contents of the clipboard. For example, if you copy the text **"Hello!"** to the clipboard and then run the following script:

MsgBox, 0, Example, The contents of the clipboard is: %clipboard%

The text "**The contents of the clipboard is: Hello!**" will be output to the screen.

Similarly, if you want to send text to the clipboard you can use the <u>SetEnv</u> command. E.g. To put the text "**I'm some pasted text**" into the clipboard you can run:

SetEnv, clipboard, I'm some pasted text

Special Variables

There are a number of special variables in AutoIt these are described below:

| AutoIt Variable | Contains |
|------------------|--|
| A_OSTYPE | The type of Operating System being run. Either "WIN32_WINDOWS" or "WIN32_NT". |
| A_OSVERSION | "WIN_XP", "WIN_2000", "WIN_NT4", "WIN_95", "WIN_98", "WIN_ME". |
| A_SCRIPTNAME | The current script name, e.g. "myscript.aut". |
| A_SCRIPTDIR | The directory that the current script resides in. |
| A_SCRIPTFULLPATH | The full pathname of the current script, e.g. "C:\Windows\Scripts\myscript.aut". |
| A_WORKINGDIR | The current working directory. |
| A_NUMBATCHLINES | The current setting of the "SetBatchLines" command. |
| A CEC | 0. 1. 6. 1. 1. (0. 50) |
| A_SEC | Seconds after the minute (00-59) |
| A_MIN | Minutes after the hour (00-59) |
| A_HOUR | Hours after midnight (00-23) |
| A_MDAY | Day of the month (01-31) |
| A_MON | Month of the year (01-12) |
| A_YEAR | The current year (4 digit format, e.g. 2000) |
| A_WDAY | Day of the week (1-7) |
| A_YDAY | Day of the year (1-366) |

AdlibOn AdlibOff

This will enable/diable processing of the [ADLIB] section.

BlockInput, <on | off>

This command with disable both mouse and keyboard input. This only works on various operating system configurations as indicated below:

| Operating System | "BlockInput" Results |
|---|--|
| | |
| Windows 95 | No effect. |
| Windows 98 | User input is blocked but AutoIt is also unable to simulate input. |
| Windows NT 4 (<i>Without</i> ServicePack 6) | No effect. |
| Windows NT 4 (<i>With</i> ServicePack 6) | User input is blocked and AutoIt can simulate input. |
| Windows 2000 | User input is blocked and AutoIt can simulate input. |

Input will be automatically enabled when the script closes.

Break, <on | off>

This command tells AutoIt whether or not the user can close AutoIt. Default is "on" (i.e. user can close AutoIt)

e.g. Break, On

DetectHiddenText, <on | off>

Some programs use hidden windows and hidden text on windows (e.g. Backup Exec) this can cause problems when trying to script them. This command allows you to tell AutoIt whether or not to detect this hidden text. Default is "off".

Exit [,<exit code>]

This command will end the script. If the optional <exit code> is used, this will return the numeric exit code to the calling process. This can be used in DOS batch files like so:

@ECHO OFF
AutoIt.exe myscript.aut
ECHO Exit code of AutoIt was %ERRORLEVEL%

EnvAdd, <Variable>, <Value>
EnvSub, <Variable>, <Value>
EnvMult, <Variable>, <Value>
EnvDiv, <Variable>, <Value>

Allows you to add, subtract, multiply and divide with Env variables. If the variable or value is not numeric, it will be taken to be zero (0). Note this functions perfrom only integer (whole number) functions - any remainders are dropped.

e.g.SetEnv, test, 20EnvAdd, test, 20

MsgBox, 0, Example, %test%

This will output "40" to the screen.

FileAppend, <Text>, <Filename>

This command will append "Text" to the end of a file. If the file does not exist, it will be created.

e.g.

FileAppend, This is line 1\nThis is line 2\n, myfile.txt

This will append two lines of text to "myfile.txt", note the use of "\n" to indicate a newline is required.

If the command is successful %ERRORLEVEL% is set to **0**, otherwise it is set to **1**.

FileCopy, <Source>, <Destination>, [1 | 0]

This command will copy a file or files from <Source> to <Destination>. Simple wildcards (*) are supported. If the optional last parameter is **1** then existing files are overwritten.

e.g.

FileCopy, C:*.exe, C:\\TestDir*.exe

If the command is successful %ERRORLEVEL% is set to **0**, otherwise it is set to **1**.

FileCreateDir, < Directory>

This command will create the directory < Directory >.

e.g.

FileCreateDir, C:\\TestDir

If the command is successful %ERRORLEVEL% is set to **0**, otherwise it is set to **1**.

FileDelete, <File>

This command will delete the specified <File>. Wildcards are supported.

e.g.

FileDelete, C:\\Test*.aut

This will delete all files called *.aut in the C:\Test directory.

If the command is successful %ERRORLEVEL% is set to **0**, otherwise it is set to **1**.

FileInstall, <Source>, <Destination>, [1 | 0]

This command is a directive for the Aut2Exe compiler and allows you to add extra files to the resulting compiled script and to copy them to the disk during execution of the script.

The file <Source> is added **during script compilation**. When the compiled script is executed and the same "FileInstall" command has been reached, the file is then extracted to <Destination>. The last optional parameter can be set to **1** to indicate that existing files should be overwritten.

Files added to a script are compressed and also encrypted.

If this command is used in an normal (uncompiled) script, a simple file copy will be performed instead -- this will help the testing of scripts that will eventually be compiled.

Note: Wildcards are **not** supported.

e.g.

FileInstall, C:\\atool.exe, %TEMP%\\atool.exe

If the command is successful %ERRORLEVEL% is set to **0**, otherwise it is set to **1**.

FileReadLine, <Variable>, <Filename>, <Line>

This command will read a line from a file of text into a variable. Line 1 is taken to be the first line.

e.g.

FileReadLine, test, myfile.txt, 1

Will read the first line of "myfile.txt" into the %test% variable.

If the command is successful %ERRORLEVEL% is set to **0**, otherwise it is set to **1**.

FileRemoveDir, < Directory>

This command will delete the directory <Directory>. Note: the directory must be empty.

e.g.

FileRemoveDir, C:\\TestDir

If the command is successful %ERRORLEVEL% is set to **0**, otherwise it is set to **1**.

FileSelectFile, <Filename Var>, <1 | 0> [,<Working Directory>]

This command will show the standard file selection dialog. If the second parameter is "1" then the filename must already exist. On completion, %ERRORLEVEL% is set to **0** if successful. If the user cancels to operation %ERRORLEVEL% is set to **1**.

e.g. To prompt the user for a filename that already exists FileSelectFile, filenamevar, 1 MsgBox, 0, Example, Filename selected was %filenamevar%

e.g. To prompt the user for a filename that already exists, and initially open in the "C:\My Files" directory
FileSelectFile, filenamevar, 1, C:\My Files
MsgBox, 0, Example, Filename selected was %filenamevar%

Gosub, < label > Return

Just like the old BASIC commands. Gosub will branch to a specified label. Return will return to the line after the Gosub command.

e.g.
SetEnv, MyMessage, This is the first message!
Gosub, mysubroutine
SetEnv, MyMessage, This is the second message!
Gosub, mysubroutine

SetEnv, MyMessage, This is the third message! Gosub, mysubroutine Exit

mysubroutine:
MsgBox, 0, AutoIt, %MyMessage%
Return

Goto, < label>

Script execution will continue at the specified label.

e.g.; This script will loop forever myloop:
Sleep, 10
Goto, myloop

HideAutoItDebug, <on | off>

This command can be used to completely hide the AutoIt script output to the AutoIt window during execution.

e.g. HideAutoItDebug, Off

HideAutoItWin, <on | off>

This command can be used to completely hide the AutoIt window and tray icon from the user.

<u>IfInString</u>, <<u>String Variable</u>>, <<u>Search String</u>>, <<u>Command</u>> <u>IfNotInString</u>, <<u>String Variable</u>>, <<u>Search String</u>>, <<u>Command</u>>

Checks to see if the contents of the <Search String Variable> is/is not present in the string <String>. If the condition is met, <Command> will be executed.

If a match was made, %ERRORLEVEL% will be set to **0**. Otherwise, %ERRORLEVEL% will be set to **1**.

e.g. To check if the word "Hello" is in the string "Hello this is a string"

SetEnv, search, Hello this is a string IfInString, search, Hello, MsgBox, 0, Matched, Yes, the search string was found

(This will output "Yes, the search string was found)

IfWinExist, <Window Title>, [<Window Text>], <Command>
IfWinNotExist ,<Window Title>, [<Window Text>], <Command>
IfWinActive, <Window Title>, [<Window Text>], <Command>
IfWinNotActive, <Window Title>, [<Window Text>],

<Command>

If the condition is met (i.e. the window title and text exists, is active, doesn't exist, is inactive) then the script will execute the <Command>.

N.B. If no window text is to be given you MUST include the extra comma.

e.g.

IfWinExist, Win Title, Win Text, Goto, label1 IfWinExist, Win Title,, Goto, label1

<u>IfEqual, <variable>, <text>, <Command></u> <u>IfNotEqual, <variable>, <text>, <Command></u>

Will compare the DOS variable <variable> with <text> and branch depending on the result.

e.g.

InputBox, MyVar, AutoIt, Please enter the word: WOMBLE IfEqual, MyVar, WOMBLE, Goto, match MsgBox, 0, AutoIt, You didnt type the correct word! Exit

match:

MsgBox, 0, AutoIt, Well done! You typed in the correct word!

IfGreater, <variable>, <number>, <Command>
IfGreaterOrEqual, <variable>, <number>, <Command>
IfLess, <variable>, <number>, <Command>
IfLessOrEqual, <variable>, <number>, <Command>

Will compare the DOS variable <variable> with <number> and branch depending on the result. If the parameters are not numeric they will be treated as being zero.

IfExist,<file or directory>,<Command>

IfNotExist, < file or directory > , < Command >

Will cause the script to execute <Command> if the specified file or directory exists/doesn't exists.

e.g.
IfExist, C:\\COMMAND.COM, Goto, fileexist
Exit
fileexist:

; Script will get to here if C:\COMMAND.COM DOES exist. MsgBox, 0, AutoIt, File exists

<u>IfMsgBox</u>, <<u>return value</u>>, <<u>Command</u>>

Use this to execute a command based on the button the user pressed in the MsgBox command. Valid return values from MsgBox are: ABORT, CANCEL, IGNORE, NO, OK, RETRY, YES

e.g. To put up a dialog box, with OK and CANCEL buttons, then branch if the user selects CANCEL:

; Example Script MsgBox, 1, AutoIt, This is a test message IfMsgBox, CANCEL, Goto, cancellabel Exit cancellabel:

; If the script gets to here, user pressed cancel MsgBox, 0, AutoIt, The user pressed cancel

<u>IniRead</u>, <<u>Variable</u>>, <<u>Filename</u>>, <<u>Section</u>>, <<u>Key</u>>
<u>IniWrite</u>, <<u>Value</u>>, <<u>Filename</u>>, <<u>Section</u>>, <<u>Key</u>>
<u>IniDelete</u>, <<u>Filename</u>>, <<u>Section</u>>, <<u>Key</u>>

Allows reading and writing of standard windows .ini files., i.e. the following format.

[SectionName] KeyName=Value

e.g. To read the value of "mykey" from the "mysection" part of an ini file, into the variable "%result%": IniRead, result, c:\\mypath\\myfile.ini, mysection, mykey

e.g. To write the text "womble" into the same section and key: IniWrite, womble, c:\\mypath\\myfile.ini, mysection, mykey

e.g. To write the DOS path information into the same section and key: IniWrite, %path%, c:\\mypath\\myfile.ini, mysection, mykey

e.g. To delete the DOS path information into the same section and key:

IniDelete, c:\\mypath\\myfile.ini, mysection, mykey

NB. The full path and filename of the .ini file must be given.

InputBox,<variable>, <title>, <message> [,hide]

Will cause a dialog box with <message> to appear. The user can enter text, press OK, and the text will be stored in the DOS variable <variable>. If the "hide" parameter is used, input will be masked (eg. for passwords).

e.g.

InputBox, MyMessage, AutoIt, Please - Enter some text to display! MsgBox, 0, AutoIt, %MyMessage%

e.g.

InputBox, password, AutoIt, Please enter your password (input is hidden), hide

Simulates a left or right mouse button click. The X and Y co-ordinates are relative to the currently active window. Run AutoIt in reveal mode to determine the required co-ordinates of a window.

To perform a double-click, simply run the command twice:)

Drags the mouse pointer from x1,y1 to x2,y2 with the relevant mousebutton held down. The co-ordinates are relative to the current active window.

MouseGetPos, <x Variable>, <y Variable>

This command will get the current position of the mouse cursor into the 'x' and 'y' variables. The co-ords are relative to the active window.

e.g.

MouseMove, <x pos>, <y pos>

This command will move the mouse cursor to the position <xpos>, <ypos>. The co-ords are relative to the active window.

MsgBox, <display mode>, <title>, <message>

Displays a dialog box with the specified message. Different display modes will give different results (appearance, number of buttons). A list of modes is given below, add up the numbers of the display modes you want.

e.g. To display "Hello" with just an "OK" button: MsgBox, 0, AutoIt, Hello

e.g. To display "Hello" with an exclamation box and OK and Cancel: MsgBox, 49, AutoIt, Hello (1=OKCANCEL, 48=Exclamation, = 49)

| Function | AutoIt Value |
|---------------------|--------------|
| MB_OK | 0 |
| MB_OKCANCEL | 1 |
| MB_ABORTRETRYIGNORE | 2 |
| MB_YESNOCANCEL | 3 |
| MB_YESNO | 4 |
| MB_RETRYCANCEL | 5 |
| | |
| | |

| MB_ICONHAND | 16 |
|--------------------|------|
| MB_ICONQUESTION | 32 |
| MB_ICONEXCLAMATION | 48 |
| MB_ICONASTERISK | 64 |
| | |
| MB_APPLMODAL | 0 |
| MB_SYSTEMMODAL | 4096 |
| MB_TASKMODAL | 8192 |

Random, < Output Variable>, < Min Value>, < Max Value>

This command produces a random number between <Min Value> and <Max Value>. (These values must be between 0 and 32767).

e.g. To produce a random number between 1 and 200 Random, output, 1, 200 MsgBox, 0, Result, %output%

(This would output the random number between 1 and 200)

RegRead, Variable, ValueType, RegKey, RegSubkey, ValueName

This command allows you to read REG_SZ and REG_DWORD values from the regsitry. If the command is successful %ERRORLEVEL% is set to **0**, otherwise it is set to **1**.

RegKey must be either "HKEY_LOCAL_MACHINE", "HKEY_CURRENT_USER", "HKEY_USERS", "HKEY_CURRENT_CONFIG" or "HKEY_CLASSES_ROOT".

ValueType must be "REG_DWORD" or "REG_SZ".

e.g. To read the location of the "Program Files" directory into the variable "TestKey":

RegRead, TestKey, REG_SZ, HKEY_LOCAL_MACHINE, Software\\Microsoft\\Windows\\CurrentVersion, ProgramFilesDir MsgBox, 0, Example, Program Files are located in %TestKey%

RegWrite, ValueType, RegKey, RegSubKey, ValueName, Value

Similar to the RegRead command, this command will allow you to create or modify a registry key. If the command is successful %ERRORLEVEL% is set to **0**, otherwise it is set to **1**.

e.g. To change the wallpaper of the current user:
RegWrite, REG_SZ, HKEY_CURRENT_USER, Control
Panel\\Desktop, Wallpaper, C:\\Mycrazybitmap.bmp

RegDelete, RegKey, RegSubKey, ValueName

This command will delete a regsitry **value**. If the command is successful %ERRORLEVEL% is set to **0**, otherwise it is set to **1**.

e.g. To delete the wallpaper value of the current user (not particularly recommended :)):

RegDelete, HKEY_CURRENT_USER, Control Panel\\Desktop, Wallpaper

Run, <Program path and name> [,<Working directory> [,<max | min | hide>]]

Executes a given program and proceeds to the next line of the script. N.B. The program to run and the working directory are separated by a comma ','. The working directory is optional.

>>>> N.B. Because the '\' character is special you must use '\\' when specifying paths! <<<<

e.g.

Run, notepad.exe, C:\\WINDOWS

Run, C:\\Program Files\\Microsoft Office\\Office\\WinWord.exe

The first command runs notepad and sets the working directory to "C:\WINDOWS".

The second command runs MS Word in the current directory.

You can also run command interpreter commands line Echo, Copy, etc.

e.g.

Run, COMMAND.COM /C Echo Hello > C:\\Hello.txt

Run, %COMSPEC% /C Echo Hello > C:\\Hello.txt

Creates a file called C:\Hello.txt containing the word "Hello".

RunWait, <Program path and name> [,<Working directory> [, <max | min | hide>]]

Same as the Run command but waits for the program to finish before continuing (recommended when running DOS commands such as copy, md, del, etc.).

This command will also set the variable %ERRORLEVEL% to the return code of the program.

Repeat, <Count> EndRepeat

This will repeat a section of the script up to the following "EndRepeat" command a total of <Count> times. If <Count> is zero, the loop will be infinite. Repeat statements can be nested.

e.g.Repeat, 10MsgBox, 0, Example, This will be output ten times!EndRepeat

Send, <Series of key presses>

Sends a set of keystrokes to the currently active window. (The syntax of these keystrokes can be found in the following section).

e.g
Send, This is a line of text#{ENTER}

SetCapslockState, <on | off>

This command will correctly set the state of the CAPSLOCK key to either **on** or **off**.

e.g. To turn on the CAPSLOCK key SetCapslockState, on

SetEnv, <variable>, <value>

This command will set an environment variable to the specified value. N.B. This variable only exists within AutoIt you will not be able to access it from DOS.

e.g. To set the Env variable "ERROR" to "There has been an error": SetEnv, ERROR, There has been an error.

To use this variable, use the percent symbol:

e.g. After running the previous command, this will output "There has been an error":

MsgBox, 1, AutoIt, %ERROR%

SetBatchLines, < lines>

Usually, AutoIt executes each line of a script during a 10ms timer cycle (under NT is it 10ms, under 9x it could be a much larger time). This can mean that in scripts which do a lot of String/Variable processing, the scripts can be quite slow. The SetBatchLines command can be used to change the number of script lines that are processed in a timer cycle. The default is 1. The maximum value is 32767. Note: For very high values, AutoIt will start to impact the available CPU time. The current value of SetBatchLines can be obtained from the special variable "A_NUMBATCHLINES".

Note: Most users will not need to uses this command at all!

e.g. To execute 100 scripts lines per cycle SetBatchLines, 100

<u>SetKeyDelay</u>, <milliseconds>

Changes the delay between keystrokes in milliseconds. Max is 32767. Default is 20ms. 1000 milliseconds = 1 second.

SetStoreCapslockMode, <on | off>

By default, at the start of a "Send" command AutoIt will store the state of the CAPSLOCK key; at the end of the "Send" command this status will be restored. Use this command to turn off this behavior.

e.g. To prevent AutoIt from modifying the state of the CAPSLOCK key during "Send" commands
SetStoreCapslockMode, off

<u>SetTitleMatchMode</u>, < mode>

Changes the way that window titles are matched against the script. Valid modes are "1" and "2" - the default is "1". This mode affects most of the commands within AutoIt that have any sort of window title and text in the command, i.e. WinWait, WinWaitActive, IfWinActive, etc.

e.g.; Script using mode 1 (default)SetTitleMatchMode, 1

; Script using mode 2 SetTitleMatchMode, 2

mode 1

In the script you specify the start of a window title to match. i.e. for the notepad.exe window (Untitled - Notepad), valid matches would be:

"Untitled", "Untitled -", "Unt" and "Untitled - Notepad".

mode 2

In the script you can specify ANY substring of the window title you want to match. Again for the notepad.exe window valid matches would be:

"Untitled", "Untitled - Notepad", "Notepad", "No".

SetWinDelay, <Milliseconds>

This changes the time that AutoIt pauses after carrying out a window related function (waiting for, minimizing, restoring, etc.) before continuing. This is useful on very slow machines, or when you have things like window animation enabled. The default is 500 milliseconds.

e.g.
SetWinDelay, 2000
Run, Notepad.exe
WinWaitActive, Untitled - Notepad
Send, Hello

In this example there would be a pause of 2000 milliseconds (2 seconds) after notepad has started until the words "Hello" appear.

Shutdown, <Flag>

This command allows you to perform various forms of shutdown. The type is determined by <Flag>. The flag can be a combination from the table below:

| Function | Flag |
|---|------|
| Log off the current user | 0 |
| Shutdown the workstation | 1 |
| Reboot the workstation | 2 |
| Force closing of applications (may lose unsaved work) | 4 |
| Shutdown and power off (if supported) | 8 |

e.g. To shutdown and force applications to close without saving = Shutdown + Force = 1 + 4 = 5: Shutdown, 5

Sleep, <milliseconds>

Halts execution of the script for the given number of milliseconds. Max is **2147483647**.

1000 milliseconds = 1 second.

SplashTextOn, <Width>, <Height>, <Title>, <Message>

This command brings up a window the specified message and title. The window stays topmost without interfering with windows behind it. Useful for automation when you can bring up a "Don't Touch"

message for the user without affecting other windows.

SplashTextOff

Removes the splash text screen from view.

StringCaseSense, <On | Off>

This command will turn case sensitivity on or off for the commands: IfEqual, IfNotEqual, IfInString, IfNotInString, StringReplace and StringGetPos.

Default is **off** (i.e. comparisons are NOT case sensitive)

StringLeft, <Output Variable>, <Input Variable>, <Number of chars to extract>
StringRight, <Output Variable>, <Input Variable>, <Number of chars to extract>

This command takes the contents of <Input Variable>, extracts a number of characters and places the result in <Output Variable>.

e.g. To extract the leftmost 5 characters of a string
SetEnv, test, Hello this is a test string
StringLeft, output, test, 5
MsgBox, 0, AutoIt, The resulting string is %output%

(The output would be "Hello")

e.g. To extract the rightmost **6** characters of a string SetEnv, test, Hello this is a test string

StringRight, output, test, 6
MsgBox, 0, AutoIt, The resulting string is %output%

(The output would be "string")

<u>StringMid</u>, <<u>Output Variable</u>>, <<u>Input Variable</u>>, <<u>Start char</u>>, <<u>Number of chars to extract</u>>

This command takes the contents of <Input Variable>, extracts a number of characters starting from <Start char> and places the result in <Output Variable>.

e.g. To extract 4 characters starting from character 7 SetEnv, test, Hello this is a test string StringMid, output, test, 7, 4 MsgBox, 0, AutoIt, The resulting string is %output%

(The output would be "this")

StringLen, < Output Variable > , < Input Variable >

This command takes the contents of <Input Variable> and puts the number of characters in the string into <Output Variable>.

e.g.
SetEnv, test, Hello

SetEnv, test, Hello this is a test string StringLen, output, test MsgBox, 0, AutoIt, The string length is %output%

(The output would be "The string length is 27")

<u>StringReplace</u>, <<u>Output Variable</u>>, <<u>Input Variable</u>>, <<u>Search String</u>>, <<u>Replace String</u>>

This command will search for the <Search String> in the contents of the variable <Input Variable>. The search string will be replace by <Replace String> and the result will be placed in the variable <Output Variable>. If the search string cannot be found, the contents of <Output Variable> will be the same as <Input Variable>.

If a match was made, %ERRORLEVEL% will be set to **0**. Otherwise, %ERRORLEVEL% will be set to **1**.

e.g.

SetEnv, teststring, Hello this is a test string StringReplace, output, teststring, test, testing testing 1 2 3 MsgBox, 0, AutoIt, %output%

(The output would be "Hello this is a testing testing 1 2 3 string")

StringTrimLeft, <Output Variable>, <Input Variable>, <Number of chars to trim>
StringTrimRight, <Output Variable>, <Input Variable>, <Number of chars to trim>

This command takes the contents of <Input Variable>, trims a number of characters and places the result in <Output Variable>.

e.g. To trim the leftmost 6 characters of a string
SetEnv, test, Hello this is a test string
StringTrimLeft, output, test, 6
MsgBox, 0, AutoIt, The resulting string is %output%

(The output would be "this is a test string")

e.g. To trim the rightmost 7 characters of a string
SetEnv, test, Hello this is a test string
StringTrimRight, output, test, 7
MsgBox, 0, AutoIt, The resulting string is %output%

(The output would be "Hello this is a test")

<u>StringGetPos</u>, <<u>Output Variable</u>>, <<u>Input Variable</u>>, <<u>Search</u> Text>

This command takes the contents of <Input Variable>, searches for the string <Search Text> and returns the position of the string in <Output Variable>. If the search string is not found, %ERRORLEVEL% is set to **1**, otherwise it is set to **0**. Position "0" is the first character.

e.g. To get the position character of a string
SetEnv, test, Hello this is a test string
StringGetPos, output, test, this
MsgBox, 0, AutoIt, The search string is at position %output%

(The output would be "The search string is at position 6")

WinGetActiveStats, <Title Var>, <Width Var>, <Height Var>, <Xpos Var>, <Ypos Var>

This command allows you to receive much information from the active window, including: Windows title, window width, window height, window x-position and window y-position.

*e.g.*Sleep, 2000
WinGetActiveStats, titlevar, widthvar, heightvar, xposvar, yposvar

MsgBox, 0, AutoIt, Title=%titlevar% Width=%widthvar% Height=%heightvar% Position=%xposvar%,%yposvar%

WinGetActiveTitle, <Variable>

This command gets the title of the active window and puts it in the DOS variable < Variable >

e.g.Sleep, 2000WinGetActiveTitle, myvarMsgBox, 0, AutoIt, The active window was %myvar%

WinKill, <Window Title> [,<Window Text>]

Similar to WinClose, but if the window has not closed after a couple of seconds (Asking to save, crashed, etc.) the window will be forcibly terminated.

WinWait, <Window Title> [,<Window Text> [,<Timeout>]]

Stops script execution until the specified window exists. The window does not need be active.

If specified the "Timeout" is in seconds. After the line has executed the Env variable %ERRORLEVEL% will be set to **0** if the function completed normally, or **1** if the wait timed out. The maximum value of Timeout is **32767** seconds.

e.g. Wait forever for the notepad window WinWait, Untitled - Notepad

e.g. Wait for five seconds for the notepad window WinWait, Untitled - Notepad,, 5

WinWaitClose, <Window Title> [,<Window Text> [,<Timeout>]]

Stops script execution until the specified window ceases to exist.

If specified the "Timeout" is in seconds. After the line has executed the Env variable %ERRORLEVEL% will be set to **0** if the function completed normally, or **1** if the wait timed out.

e.g. WinWaitClose, Untitled - Notepad

WinWaitActive, <Window Title> [,<Window Text> [,<Timeout>]]

Stops script execution until the specified window to exists and is active.

If specified the "Timeout" is in seconds. After the line has executed the Env variable %ERRORLEVEL% will be set to **0** if the function completed normally, or **1** if the wait timed out.

WinWaitNotActive, <Window Title> [,<Window Text> [, <Timeout>]]

Stops script execution until the specified window ceases to be active.

If specified the "Timeout" is in seconds. After the line has executed the Env variable %ERRORLEVEL% will be set to **0** if the function

WinHide, <Window Title> [,<Window Text>]
WinShow, <Window Title> [,<Window Text>]
WinRestore, <Window Title> [,<Window Text>]
WinMinimize, <Window Title> [,<Window Text>]
WinMaximize, <Window Title> [,<Window Text>]
WinActivate, <Window Title> [,<Window Text>]
WinClose, <Window Title> [,<Window Text>]

If the window exists, the relevant command (i.e. minimizing, hiding, activating, etc.) will be performed.

WinMove, <Window Title>, [<Window Text>], <X>, <Y>, <Width>, <Height>

If the window in Title and Text exists, it is moved to <X>, <Y> and resized to <Width>, <Height>

e.g.

Run, notepad.exe WinWaitActive, Untitled - Notepad WinMove, Untitled - Notepad,, 0, 0, 100, 100

This moves the notepad window to 0,0 and changes the size of the window to 100,100

Remember to include the comma if you don't specify Window text. "default" can be used in place of Width and Height if you don't want to change the size of the window

WinSetTitle, <Window Title>, [Window Text], <New Title>

Use this command to rename a window.

e.g.

Run, notepad.exe WinWaitActive, Untitled - Notepad WinSetTitle, Untitled - Notepad,, Renamed Notepad!

WinMinimizeAll WinMinimizeAllUndo

The "WinMinimizeAll" command minimizes all windows; "WinMinimizeAllUndo" will undo this action. If any windows are manipulated by AutoIt or the user, then "WinMinimizeAllUndo" will not have any effect on these "touched" windows.

Script Compiler

AutoIt comes with two programs to enable the complication of scripts into standalone executables (and back again if required).

From v2.64 the facility to add extra files into the compiled executable has been added via the FileInstall command.

Aut2Exe - Script Compiler

This program will take an AutoIt script and convert it to an executable. This executable can then be run without any of the original AutoIt programs being installed.

It is possible to define an icon for the resulting executable, this must be an icon file that is 32 by 32 pixels in 16 colours (the file should have a filesize of 766 bytes).

Scripts and files added to a compiled executable are compressed and then encrypted. During execution of the compiled script the script is **never** written to disk -- all decompilation of the script is performed in memory for security.

N.B. You can still pass command line variables to the resulting exe file by running "myscript.exe param1 param2.....".

Command Line Switches

Aut2Exe can be used with command line switches like so (all

switches are optional except "/in"):

Aut2Exe.exe /in <script.aut> [/out <script.exe>] [/icon <icon.ico>] [/pass <passphrase>]

Long filenames (and passphrases that contain spaces) must be enclosed in quotes, for example:

Aut2Exe.exe /in "C:\my programs\my script.aut" /out "script.exe" /pass "This is my secret"

Note: Where an icon is not specified on the command line, the last icon that was used to compile a script is used by default.

Note: If a destination file is not specified, the resulting file will be called the same as the input file, but with the extension changed to ".exe". For example, "myfile.aut" would be compiled by default to "myfile.exe".

Exe2Aut - Script Decompiler

This is a utility that will convert and previously compiled .exe back into an AutoIt script file. If a passphrase was used to compile the script, you will be required to enter the same passphrase to decompile the .exe. (We wouldn't want people decompiling other peoples scripts now, would we?:))

Command Line Switches

Command line switches are similar to those of Aut2Exe:

Exe2Aut.exe /in <script.exe> [/out <script.aut>] [/pass <passphrase>]

Frequently Asked Questions (FAQ)

There is an updated FAQ online at this address:

The ADLIB section.

- Does the Exit command work in the AdLib section?
- : Can a "GoSub" be used in an AdLib statement?

Check, Dialog, and Input Boxes

- Does anybody have a way of finding out whether a checkbox is currently selected?
- <u>Is it possible to have a message window pop up, type a value, then have that value re-typed when called?</u>
- How do I take a user ID and password from Input Boxes and later send it to another application?

Directory Management

- How do I create a directory from within AutoIt?
- How do I delete a directory from within AutoIt?
- How do I delete all of the files in a directory?

Disk Management

- How do I format a bootable system diskette from within AutoIt?
- How can I defrag my hard drive on a nightly basis using a W2K scheduler?

Display Manipulation

- How do I run an external command dependent on the display resolution?
- How do I save a screen or single window for later scrutiny?

General AutoIt Questions

- <u>Does AutoIt record keystrokes/mouse clicks and create a script or do I need to write it manually?</u>
- <u>Is there a way to have autoIt randomly select a file out of a directory or list of files?</u>
- How do I empty the recycle bin from within AutoIt?
- What exactly does "Reveal mode" do?
- How can I schedule the running of an AutoIt script?
- Does anyone know if an AutoIt archive exists?
- Has anyone used AutoItX within a Visual Basic Application?

General Windows Questions

- Is there a sure way to command my Windows to reboot?
- What do I do if other users are connected to the system as well?
- How do I shutdown a server down when users are connected?
- How do I shutdown all PC's on a local network by running a script from a Windows 98 "server"?
- Is it possible to send keystrokes to invisible windows with AutoIt?
- <u>Is their anyway to have AutoIt hold a mouse click down (for a second or so)?</u>
- Is it possible to bring the cursor back to a predefined position after a mouse movement command?

- Is there a way to determine that Windows has finished booting?
- How do I create a shortcut from within AutoIt?
- How can I detect a window with varying titles?

Networking

- <u>Is there an easy way to change the Gateway IP (in TCP/IP) for a network card using AutoIt?</u>
- How do I run an AutoIt script at or before the logon screen in Windows NT?
- Can I use AutoIt to login a user on Windows NT or 2000

Registry Manipulation

- How do I save an uninstall string value in a variable, and run it?
- How can I read/write default entries to/from the registry?
- How do I delete a registry key?
- How do I delete a registry key without using an external command? What can I write in a .REG file?
- Is there a way from AutoIt to secure a Registry key?

The RUN and RUNWAIT commands and the COMSPEC variable

- How do I run an external command?
- <u>Is there a way to activate the Run box (Winkey-R) dialogue</u> <u>quickly?</u>
- How do I play a WAV file from within AutoIt?
- Why doesn't the DOS command window close automatically?

The Send Command

- How do I output spaces that are at the beginning of a line?
- With AutoIT can I work with excel cells?

Special Characters

Backslashes

The SplashText Command

- How do I move the SplashText display box?
- Can a picture be put in the SplashText window?

The String Manipulation Commands

- How do I remove spaces from a string?
- What am I doing wrong with the StringLeft command?
- <u>Is there a way to return the position of a substring within a string?</u>
- <u>Is there a difference between reading a line with a blank space</u> and carriage return vs. just a carriage return?

Variable Resolution

- How does variable resolution work in AutoIt?
- How do you export an AutoIt variable to DOS?

History

17/12/2002 - v2.64

- More Windows eys options added to the <u>Send</u> command: LWINDOWN, RWINDOWN.
- Added command: <u>HideAutoItDebug</u> to turn off the script being shown during execution.
- Internal code optimisations and modularisation.
- Full installation adds registry keys for exe locations: HKLM\Software\HiddenSoft\...\InstallDir
- Added simple file operation commands: <u>FileCopy</u>, <u>FileDelete</u>, <u>FileCreateDir</u>, <u>FileRemoveDir</u>.
- Added new compiler directive <u>FileInstall</u> for compressing and adding extra files to a compiled script.

04/02/2002 - v2.63

• WinActivate improved under 2000/XP (Hopefully!).

19/01/2002 - v2.62

- Passphrase added to Aut2Exe.
- Exe2Aut added for script decompilation.
- Various bugs fixed with <u>SetBatchLines</u>.
- Some division-by-zero bugs fixed.

26/08/2001 - v2.61

 CAPSLOCK behaviour changed. The state of the CAPSLOCK key is checked and stored at the start of a "send" command and restored at the end of a "Send" command (you can still send CAPSLOCK keys from within a "Send" command but remember that it is only in effect for that single "Send" command). Previous behaviour was that AutoIt turned off CAPSLOCK at **startup** which could lead to unexpected results with users and the CAPS keys.

- Added command: <u>SetStoreCapslockMode</u> to turn off **any** CAPSLOCK modification as described above.
- Added command: <u>SetCapslockState</u> to correctly allow setting of the CAPSLOCK to **on** or **off**.
- Bug with SetBatchLines and RunWait fixed.
- Bug with allowing division by zero in EnvDiv and Random fixed.
- Installation package recompiled hopefully fixing installation bugs under Win ME.

14/05/2001 - v2.60

- AutoIt and Aut2Exe re-coded in C++. Thanks to the beta testers for their help in making sure I didn't break everything.
- Added commands: <u>EnvMult</u>, <u>EnvDiv</u>, <u>WinMinimizeAllUndo</u>, <u>StringGetPos</u>, <u>SetBatchLines</u>, <u>WinGetActiveStats</u>, <u>FileSelectFile</u>.
- Added command: #include.
- Operating system version <u>%A_OSVERSION</u>% variable added (WIN_XP, WIN_2000, WIN_NT4, WIN_95, WIN_98, WIN_ME).
- StringReplace and IfInstring now set the ERRORLEVEL to 0 if a match was made otherwise ERRORLEVEL set to 1.
- Stopped AutoIt detecting it's own window during WinWait type commands.
- Maxium characters in a script line increased to 16384 (should enable larger clipboard related actions).
- Command line parameters and switches changed for both <u>AutoIt</u> and <u>Aut2Exe</u>.
- Error messages improved. Line numbers and the line with the error is now displayed.

- The variable %0% contains the number of <u>command line</u> parameters.
- Command line "/c" option added. Allows the running of a single script line without a script, e.g. Autoit.exe /c "WinWait, Untitled Notepad"
- Previous documentation is incorrect on the ranges of numeric values in AutoIt (ouch sorry). Unless specified otherwise, numeric values are 32bits wide, i.e. -2147483648 to 2147483647. But stay a little short of the maximum (2147483647) for Sleep and timeout commands as you will get unexpected results. The good news is that you can now "Sleep" for much longer periods in a single command (I'd appreciate reports on your experiences with this!).
- Some icons and logos changed as I fancied a change:)
- Optional exit code added the the "Exit" command.
- (Aut2Exe) last icon used by default.

03/12/2000 - v2.51

- Installation routine changed to use Microsoft Windows Installer.
- Added commands StringLeft, StringRight, StringMid, StringLen, StringTrimLeft, StringTrimRight, IfInString, IfNotInString, StringReplace.
- String comparisons (IfEqual, IfNotEqual, IfInString, IfNotInString, StringReplace) can have case sensitivity controlled by the StringCaseSense command.
- Added Random number command.
- Fixed bug with MsgBox and InputBox under NT4 (as a result the AutoIt window will be visible in the taskbar during these commands).
- Fixed ADLIB section so that Gosub and Goto commands will work correctly.

07/11/2000 - v2.50

- Fixed access rights with RegRead, RegWrite and RegDelete commands for non-administrator use under Windows NT.
- Added commands IfLess, IfLessOrEqual, IfGreater, IfGreaterOrEqual.
- Reveal mode includes the dimensions of the active window.

17/6/2000 - v2.42

- Added IniDelete, RegDelete, commands.
- Added the BlockInput command to disable mouse and keyboard input. (**N.B.** only works satisfactorily under Win2000/NT4 SP6 best I could do I'm afraid)
- The "/p" is now NOT case sensitive for command line parameters.
- The WinWait routines' "Timeout" is now accurate.
- Added simple file routines FileAppend and FileReadLine.
- If you use the {CTRLDOWN}, {SHIFTDOWN} or {ALTDOWN} keys, the keys will be left "down" unless specified otherwise with the {CTRLUP}, {SHIFTUP} or {ALTUP} keys (even after the end of a line in the script). Use these keys wisely:)

11/02/2000 - v2.41

• Bug with RegWrite fixed.

09/02/2000 - v2.4

- Problems with variables containing commas fixed.
- Lots of variables added to access times, dates, current script path, OS (NT or 9x).
- Can pause and resume script execution by right-clicking on the

- taskbar icon.
- SIMPLE registry read/write functions added for REG_SZ and REG_DWORD types.
- Fixed bug with command line parameters and quotes.

20/12/99 - v2.32

- Some quite brutal optimizing applied, bringing the executable sizes down by 12k.
- Aut2Exe remembers the last directory used for the script, executable and icon.
- Can specify command line parameters with spaces by using quotes ("....").

12/12/99 - v2.31

- Maximum line length increased to 1024 characters.
- Memory usage when running scripts reduced by a factor of 12! (Not that I ever had any complaints :))

10/12/99 - v2.3

- Various bugs with ENV variables fixed.
- Now sends correctly to DOS (I hope :)).
- Optional parameter added to InputBox for masking input.
- Command line parameters can be passed to the script.
- Now supports user-defined icons when compiling a script to an .exe.
- MouseGetPos and MouseMove commands added.
- Not using ASPack anymore, although feel free to use it on AutoIt yourself (http://www.aspack.com).

25/11/99 - v2.23

- WinActivate tweaked for Win2000 RC2.
- Shutdown command added.
- WinKill command added.

23/11/99 - v2.22

- The variable %CLIPBOARD% contains the textual contents of the clipboard.
- Can now paste text from AutoIt to the clipboard.
- INI File reading and writing added with IniRead and IniWrite.

20/11/99 - v2.21

• Can now Send decimal ASCII codes, like when using using the ALT key and the numeric keypad (e.g. ALT+65 = 'A'). Very useful for sending characters that don't usually work (e.g. '~' on an Italian keyboard)

18/11/99 - v2.2

- Can now Run programs as either hidden, maximized or minimized (great for avoiding black screens when executing DOS commands)
- RunWait will put a programs return code in %ERRORLEVEL%
- The WinWait group of functions can be set to timeout after a period of time (%ERRORLEVEL% can be checked to see if the function timed out or not)
- WinShow command added (previously, there was no way to "undo" a WinHide!!!!)

<u>16/09/99 - v2.11</u>

• Fixed bug in which ENV variables were not expanded under the "Adlib" section.

• Installation recreated under Win98 to resolve some setup problems.

11/09/99 - v2.1

- Aut2Exe Script converter added to installation.
- Internal bug in Run and RunWait fixed.
- Added keys {CTRLUP}, {CTRLDOWN}, {SHIFTUP}, {SHIFTDOWN}, {ALTUP}, {ALTDOWN}.

28/08/99 - v2.03

- AutoIt2.exe and AutoItX.dll now packed with ASPack reducing the executable size.
- Changed SplashTextOn command. You no longer have to do SplashTextOff before doing a second SplashTextOn.
- Added simple variable EnvAdd and EnvSub commands.
- "default" can be used in place of Width and Height in the WinMove command.
- Added commands LeftClickDrag and RightClickDrag.

26/08/99 - v2.02

- Changes to some internal timings.
- Made the splash windows like a little less "win 3.1 ish".
- Added AdlibOn and AdlibOff commands.
- Changed internal handling of the ADLIB section.

18/08/99 - v2.01

• Fixed Send command with multiple modifiers (i.e. CTRL+SHIFT+key, CTRL+ALT+key).

16/08/99 - v2.0

NB. Some changes mean that old AutoIt scripts may need to be modified slightly - Please re-read the documentation carefully if you are upgrading! Hence the jump from v1.8 to v2.0 - SORRY, but things needed changing to ease implementation of new commands that have been requested. I personally use v1.8 for old scripts and v2.0 for any new scripts I write.

- Can now send most keystrokes to DOS windows Finally!
- Changed the way in which '\', ',' and '%' are treated CAREFUL!
- Changed Run and RunWait commands. Comma ',' now delimits working directory. (therefore '*' is now valid for wildcards, etc.)
- Added numerous keys that can be sent including WinApps key.
- AutoIt window shows line number and line of script during execution.
- Helpful(?) error messages with line numbers (can't run file, invalid syntax, etc.)
- Added commands: HideAutoItWin, WinWaitNotActive, WinMove, WinSetTitle, DetectHiddenText, WinGetActiveTitle, SetWinDelay, Break, WinMinimizeAll, SplashTextOn, SplashTextOff, Repeat, EndRepeat.
- Shows script name as a tooltip on the AutoIt system tray icon.
- Following commands no longer wait for the window to exist before continuing: WinRestore, WinHide, WinMaximize, WinClose, WinMinimize.
- If run with no parameters, AutoIt will bring up a "File Open" box.
- Window in "reveal" mode always stays on top. (Now only one reveal mode required).
- Changed MsgBox command to include a title, and to not hang around in the background.
- Changed all "If...." commands to execute a given command rather

- than just jumping to a label.
- Changed InputBox to include a title, and to not hang around in the background.
- Invalid commands will now be treated as errors instead of just ignored.
- '\n', '\t' can be used in MsgBox and InputBox text to indicate newlines and tabs.
- Minimizes to a system tray icon. If the HideAutoItWin command is used nothing is visible at all.
- Changed Adlib, now executes a given command instead of just sending keys.
- Changed Sleep (internal) so that AutoIt window now updates during long Sleep functions.
- Changed Gosub/Return. Can now have nested subroutines, only limited by free memory!

02/04/99 - v1.8

- Added MsgBox and IfMsgBox commands. (Note, these have changed from 1.8 beta)
- Added IfExist and IfNotExist commands for files and directories.
- Added DOS environment variable support (e.g. Send, Path is %PATH%).
- N.B. Because of this, the percent symbol '%' is now special. If you
 - really want to use this symbol you must do it twice ie. %%
- Added SetEnv command.
- Added InputBox command.
- Added IfEqual and IfNotEqual commands.
- Added Gosub and Return commands.
- "Page-fault" bug fixed (when mouse coords are out of range).

14/03/99 - v1.7

- Added {LWIN} and {RWIN} "Window key" keystrokes.
- Added SetTitleMatchMode command. (Alters the way window titles are matched).
- A blank window title is now valid. (e.g. WinWait,, Window text)
- AutoIt executable grows for the first time to 44KB.

28/02/99 - v1.6

- Added IfWinExist, IfWinNotExist, IfWinActive, IfWinNotActive commands.
- Added support for labels (for the above commands).
- Added Goto command.
- Added Exit command.
- Window titles are now case sensitive. i.e. "NOTEPAD" and "Notepad" are different.

20/02/99 - v1.5

- Added {PRINTSCREEN} command to simulate the "PrintScrn" key.
- Copy and paste from the "reveal" mode. (Use CTRL-C to copy).
- Added command line syntax "AutoIt /?".
- Added simulated mouse clicks (left and right).
- Added mouse co-ordinates in "reveal" mode.

20/01/99 - v1.4

- Added {BS}, {INSERT}, {ESCAPE} and {DELETE}
- Added "/reveal2" command line switch. Cause AutoIt to update the reveal window every half second (useful for complex dialog windows)

18/01/99 - v1.3

- Added WinClose command.
- The title "WinWait,Notepad" will no longer match the window "Untitled Notepad", but "WinWait,Untitled" will.
- You can now use white spaces (spaces and tabs) all over the place to make the script file look pretty:)

14/01/99 - v1.2

- Changed "Run" command. Asterisk (*) now states start of working directory.
- Added [ADLIB] section support
- Officially released on winfiles.com

12/01/99 - v1.1

- Fixed WinActivate under Windows 98/NT 2000!
- Added WinMaximize
- Added executable version info
- Added icons

12/01/99 - v1.0

• Initial build (beta)

Thanks to

I've cut this section down - there are just too many people to list now! If you think that you should be on the list then let me know.

- The authors of <u>UPX</u> for their excellent file compressor used in AutoIt and AutoIt compiled scripts.
- Alex Peters for the AutoIt script icon (petersa@mail.com).
- Mark for the AutoIt and Aut2Exe Logos used prior to v2.60 of AutoIt (sdv@skanky.force9.co.uk).
- Everyone who has done a language translation for me (credits should be in their translated files :)).
- Everyone else who has since emailed me with suggestions and comments (hundreds of you!!) I hope you all continue to find AutoIt of some small use!
- Everyone who has made suggestions/comments on the AutoIt mailing lists.
- The beta testers who always manage to find new and improved ways to break AutoIt;)

```
Q #1: Does the Exit command work in the AdLib section?
A #1: No, the Exit command does not function from an ADLIB line, as in
   ; ====== Scriptlet Starts Here ======
   [ADLIB]
   Some Window, Some text, Exit
   ; ====== Scriptlet Stops Here ======
   However, the same thing can be achieved by replacing "Exit" with
   "Goto, ExitLabel" and adding the lines "ExitLabel:" and "Exit"
   similar to the following scriptlet.
   ; ====== Scriptlet Starts Here ======
   SetTitleMatchMode, 2
   AdLibOn
   SetEnv, Count, 1
   Loop:
   MsgBox, 0, The Beginning, Count = \%Count\% \n
   EnvAdd, Count, 1
   IfEqual, Count, 5, Run, notepad.exe
   IfEqual, Count, 11, Goto, ExitLabel
   Goto, Loop
   ExitLabel:
   WinWaitActive, - Notepad
   Winclose, - Notepad
   Exit
   [ADLIB]
   - Notepad, Goto, ExitLabel
   ; ====== Scriptlet Ends Here ======
```

```
Q #2: Can a "GoSub" be used in an AdLib statement?
A #1: Yes. See the script below for an example.
   ; ====== Script Starts Here ======
   SetTitleMatchMode, 2
   AdLibOn
   SetEnv, C, 1
   SetEnv, Count, 11
   Repeat, %Count%
      IfEqual, C, 5, Exit
      EnvAdd, C, 1
      Run, Notepad.exe
   EndRepeat
   Exit
   Notepad:
      Send, !fx
   Return
   [AdLib]
   - Notepad, , GoSub, Notepad
   ; ====== Script Stops Here ======
```

| Q # | #1: Does anybody have a way of finding out whether a checkbox is currently selected? toggling is easy, but a bit pointless if you don't know the initial state! |
|-----|--|
| A # | #1: You should send "+" (for enabling) or "-" (disable) to the check instead. |
| | But this is not consistant across all programs. You should test this with the specific program you are automating to make sure it works |
| | |
| Q # | #2: Is it possible to have a message window pop up, type a value, thave that value re-typed when called? I am building a tool for our accounts admin to remove all of a users accounts. I want to try to prevent them from typing in the user name value several times. |
| | #2: Is it possible to have a message window pop up, type a value, have that value re-typed when called? I am building a tool for our accounts admin to remove all of a users accounts. I want to try to prevent them from typing in the user name value several times. |
| | #2: Is it possible to have a message window pop up, type a value, have that value re-typed when called? I am building a tool for our accounts admin to remove all of a users accounts. I want to try to prevent them from typing in the user name value several times. #1: Yes, use a input box and have the value written (iniwrite) to a fitthen when you need the value have it read in from the file (inireaction). |
| | #2: Is it possible to have a message window pop up, type a value, have that value re-typed when called? I am building a tool for our accounts admin to remove all of a users accounts. I want to try to prevent them from typing in the user name value several times. #1: Yes, use a input box and have the value written (iniwrite) to a function them you need the value have it read in from the file (inireactive this: inputbox, user, Enter User Name, What is the user's name? |

| A #2: If it's all in the same script, don't bother writing to a file just do: |
|---|
| inputbox, user, Enter User Name, What is the user's name? |
| And later to send it use |
| send, %user% |
| Q #3: I need help in taking the input from an Input Message Box and whatever the input may be, writing it as a send command into an active window. In other words, I need for a user name and password to be input by the user and then write the user input to an existing application as a send function later in the script. If anyone has a script example I would greatly appreciate it. |
| A #1: try this |
| ; ====== Scriptlet Starts Here ====== InputBox,username,User Name Entry,\nPlease enter your username InputBox,password,Password Entry,\nPlease enter your password,hid Run,notepad.exe Winactivate,Untitled - Notepad Send, Username is: %username%{ENTER} Send, Password is: %password% ; ======= Scriptlet Ends Here ======= |
| |

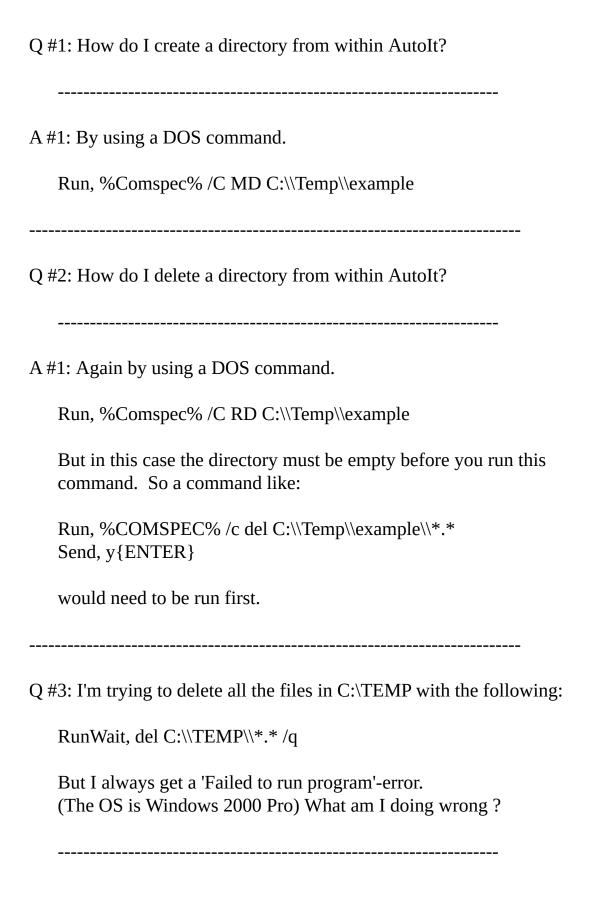
A #2: Input a value into a variable with the InputBox, then Send the variable to the application that needs it. I've been doing it for a year or so, and works almost perfectly; I say ALMOST because every

once in a while the script loses focus for whatever reason between the WinActivate command and the actual Send command, which means occassionally my Send command sends output to the desktop.

A #3: The only thing I've been able to rely on was to create my own login screen, capture and send the data as needed. I've tried spyware and password capture programs but they aren't reliable or they're too big or they're everywhere or they're something else...

```
A #4: Try using something like this:
```

```
setenv, WinTitle, <window title>
setenv, WinText, <window text>
...
setenv, keystrokes, <keystroke string>
gosub, loop
...
setenv, keystrokes, <next keystroke string>
gosub, loop
...
loop:
; ====== Scriptlet Starts Here ======
IfWinNotActive,%WinTitle%,%WinText%,Winactivate,%WinTitle%,%WifWinNotActive,%WinTitle%,%WinText%,goto, loop
Send, %keystrokes%
Return
; ======= Scriptlet Ends Here =======
```



A #1: You need something like:

runwait, %COMSPEC% /c del c:\\temp*.* /q

Because "del" is a DOS internal command. You need to specify the command shell that it is internal to. That is what the %COMSPEC% does.

Q #1: How do I format a bootable system diskette from within AutoIt? A #1: In Windows 9x you can do the following: ; ====== Scriptlet Starts Here ====== GoSub, MakeTmp FileAppend, Y\nN\n, %TMPFILE% RunWait, %COMSPEC% /c FORMAT.COM A: /S /U <%TMPFILE% RunWait, %COMSPEC% /c del %TMPFILE% Exit MakeTmp: SetBatchLines, 1000 Random, T, 1, 32767 SetEnv, TmpFile, C:\\%T%.txt SetEnv, TMPFILE, C:\\%T%.txt IfExist, %TmpFile%, GoSub, MakeTmp FileAppend, ,%TmpFile% Sleep, 1000 Send, ^Z IfExist, %TMPFILE%, Goto, MakeTmp_END SetEnv, ERRORLEVEL, 1 MakeTmp_END: SetBatchLines, 1 Return ; ====== Scriptlet Ends Here ====== Q #2: How can I defrag my hard drive on a nightly basis using a W2K scheduler? Does anyone know how to run the dfrg.msc executable in a script?

A #1: I use O&O's FreeWare Defragmenter for Win2k. Unfortunately, I removed the MS Defrag when I installed the freeware version. However, the following command in an AutoIt script allowed me to bring up the O&O Defrag program from an AutoIt script. Note: The O&O Defrag may not have command line parameters you can pass in, though MS's Defrag might.

Run, C:\\WINNT\\system32\\mmc.exe "C:\\Program Files\\OOD2KFRE\\OOD2KFRE.MSC" /s

Q #1: I'm running into a bit of a problem. What I'm wanting to do is run a program according to the display resolution. As you'll see in the script I've pasted, but what I find is that it launches all 3 of them no matter what? Probably something I've over looked!!

RegRead, TestKey, REG_SZ,

HKEY_CURRENT_CONFIG, Display\\Settings, Resolution
 IfEqual, TestKey, "1024,768", GOTO, internet
 IfEqual, TestKey, "800,600", GOTO, word
 IfEqual, TestKey, "640,480", GOTO, excel

internet:
 run, c:\\program files\\internet explorer\\iexplore.exe

word:
 run, c:\\program files\\microsoft office\\office\\winword.exe

excel:
 run, c:\\program files\\microsoft office\\office\\excel.exe

A #1: First of all, check to see the results of your RegRead by putting the Command

MsgBox, 0, RegRead Results, Result=%TestKey%

right after the RegRead, To make sure you are getting the results you think you should. Once you are satisfied with that, you can remove the MsgBox command

In addition add the goto commands I have shown here:

; ====== Scriptlet Starts Here ====== RegRead, TestKey, REG_SZ, HKEY_CURRENT_CONFIG, Display\\Settings, Resolution

```
IfEqual, TestKey, "1024,768", GOTO, internet
IfEqual, TestKey, "800,600", GOTO, word
IfEqual, TestKey, "640,480", GOTO, excel

goto, end

internet:
run, c:\\program files\\internet explorer\\iexplore.exe
goto, end

word:
run, c:\\program files\\microsoft office\\office\\winword.exe
goto,end

excel:
run, c:\\program files\\microsoft office\\office\\excel.exe

end:
; ======= Scriptlet Ends Here ======
```

Q #2: I've created a script that uses the shortcut key option in the shortcuts properties (hmm... lemme explain this another way, Rightclick on a shortcut, select properties, see 'shortcut key' you can assign hotkey for shortcuts) O.k. Not to insult anyone's intelligence, but I wanted everyone on the same page.)

This script when called upon, will take a snapshot of the currently active screen, then open mspaint.exe, paste it, and save the file in the %YEAR%%MONTH%%DAY%.... format... it gives me a screenshot of each

screen so I can go back later on and write a script that I was not able to spend the time during the initial install. So far, this method is starting to work out great, But I'd like a way to Open Autoit Reveal mode, take a snapshot of that window, saving it, then use it for later review for script making.

Problem, you can not 'WinGetAtiveTitle' it, 'WinSetTitle', or any other similar steps. All Autoit based windows stay.... Autoit, or Aut2Exe. Does anyone know of a way to do this? is this in the plans for future versions? I can not get a snapshot (screenshot using print-screen key) of the Autoit window, which has all the text listed and mouse coordinates, etc... By being able to do this, a script can be written at a later time, that would work fine, as all info on the AutoIt screen is intact, but the inability to printscreen the AutoIt window prevents this cool usage.

A #1: I use this code to capture error messages and printscreen them. Open two AutoIt reveal windows, to get your exact AutoIt version and Window Title.

```
; ====== Scriptlet Starts Here ======
SetTitleMatchMode, 2
WinActivate, AutoIt v2.41 - (C)2000 Jonathan Bennett.
WinWaitActive, AutoIt v2.41 - (C)2000 Jonathan Bennett.
SEND, #{ALTDOWN}#{PRINTSCREEN}#{ALTUP}
SLEEP, 1000
WinMinimize, AutoIt v2.41 - (C)2000 Jonathan Bennett.
WinMaximize, untitled - Paint
WinWaitActive, untitled - Paint
SLEEP, 2000
SEND, #{CTRLDOWN}#v#{CTRLUP}#{ENTER}
; ====== Scriptlet Ends Here =======
```

A #2: Try using: send, {PRINTSCREEN} rather than: send, !{PRINTSCREEN}

that should capture the whole screen, not just the active window. That would get you the application's and the autoit reveal window....

|) #1: Does AutoIt record my key strokes and create a script or do I nee | þ |
|---|---|
| to create a script manually in notepad? | |
| | |
| | |
| | |

A #1: AutoIt does not currently have a recording feature. If you want to automate some installation or other set of actions, you must learn the exact sequence of keystrokes you wish to send and write it (manually) into a script. See the Send command and the section that describes the keystroke codes (for example, {ENTER},

{ESCAPE}, {DOWN}, {TAB}, etc.) in the AutoIt documentation.

A #2: I recommend not useing Notepad, it's a really poor editor for writing any type of script. There are many other editors, like UltraEdit, TextPad, NoteTab, etc. which are much better tools for your task. Here are two editors I could recommend for newbies:

1. AutoIt editor

http://www.lkei.com/downloads/downloadsfreewarerequest.asp?2 or

- 2. NoteTab Light (freeware)
 - a) download http://www.notetab.ch/ftp/ntfree.zip
 - b) install NoteTab Light
 - c) download

http://groups.yahoo.com/group/AutoItList/files/misc/notetab.zip

d) put the uncompressed file into NoteTabs library path (usually C:\Program Files\NoteTab Light\libraries)

| Q #1: Has anyone thought of a way to have autoIt randomly select a fi | le |
|---|----|
| out of a directory or list of files? I'd like to use it as my | |
| wallpaper manager. | |

A #1: Let AutoIt open up the directory in Explorer with all the pictures there, then:

```
; ====== Scriptlet Starts Here ======
Random,numtimes,1,25

;for example of how many times to
Repeat,%numtimes%
Send, {RIGHT}
; moves selection numtimes to a file and numtimes is random
; number between 1 and 25
EndRepeat
; ======= Scriptlet Ends Here =======
```

From here you can have the pic selected in explorer and do the rest. There are more efficient ways of doing this ... be creative

| Q #1: How do I empty the recycle bin from within AutoIt? Maybe a Rundll command? |
|---|
| A #1: You can use an AutoIt script to clear it from the desktop like this: |
| ; ====== Scriptlet Starts Here ====== WinMinimizeAll leftclick,0,0 send, recy+{F10} send, b{ENTER} winwaitactive, Confirm send, !y WinMinimizeAllUndo ; ====== Scriptlet Ends Here ====== |
| It can be put in either an AutoIt script file or compiled into an EXE. If it's compiled. It becomes a lot like number 4. |
| A #2: You can use a 3rd party command such as ERB. Found at: |
| http://hjem.get2net.dk/fec/software/erb/index.html |
| This command was specially written to empty the recycle bin. It can be run from say a network drive (t:) from within AutoIt by using: |
| Run, %comspec% /c t:\\erb.exe /hide |
| It can also be run from the logon script with the following command: |
| erb.exe /hide |
| |

; ====== Scriptlet Starts Here ====== Run,C:\\\\Windows\\\\Explorer.exe \\\\Recycled Send,!fb{ENTER}!fc ; ====== Scriptlet Ends Here ====== A #4: You can run a batch file (call it say: "eraserb.bat") with the following commands in it. ; ====== Batch File Starts Here ====== C: CD\RECYCLED ATTRIB -h *.* ATTRIB +h desktop.ini ECHO Y | DEL *.* >> NUL: ; ====== Batch File Ends Here ====== A #5: You can use a single DOS internal command like: erase C:\recycled or an external DOS command like: deltree /y c:\recycled\ These commands can be run from within AutoIt like this: ; ====== Scriptlet Starts Here ====== Run, %comspec% /c deltree /y c:\\recycled\\ Run, %comspec% /c erase C:\\recycled ; ====== Scriptlet Ends Here ======

A #3: You can use Windows explorer and AutoIt to do the job like this:

A #6: Some other suggestions can be found on techrepublic.com in an article by Bill Shadish (which is where I got the batch file from). It states:

"You've probably heard the Paul Simon song "Fifty Ways to Leave Your Lover," which describes a number of tricks for breaking free from an Undesired relationship. This article places a spin on the concepts used in that song. No, I can't guarantee that simply by reading this article youll meet that special person. But This article will provide 40 ways to solve some common disk-space problems under Windows 9x and Windows NT.

Cleanup

You can use a small program to empty the Recycle Bin (29) (its really just A special directory on the root of the C drive). The C header code to do So looks like this:

```
SHEmptyRecycleBin
SHSTDAPI SHEmptyRecycleBin(
HWND hwnd,
LPCTSTR pszRootPath,
DWORD dwFlags
);
```

However, you must install the Explorer 4.0x extensions (such as channels) To get the required version 4.71 of the Shell32.dll file to make the call. I, for one, am not interested in installing tons (read that, megabytes) of glut just to be able to delete files from one directory. So, lets look at a batch file that does the same thing. Note that the recycle directory contains two special, hidden files. Info maps the names of files in the Recycle Bin to their original filenames; you should delete this file. The shell uses Desktop.ini to recognize that the Recycle Bin is a special folder; dont remove this file. (30) Place the following lines within a batch file called, perhaps, Dump.bat:

C:

CD\RECYCLED
ATTRIB -h *.*
ATTRIB +h desktop.ini
ECHO Y | DEL *.* >> NUL:

You can now empty the Recycle Bin by calling this file."

A #7: One possible solution has not been proven or necessarily shown to be a solution as yet. But the comments have been included here for completeness. It entails using the Rundll command and the shell32.dll function "SHEmptyRecycleBin". the gist is to use a command like:

rundll32 shell32, SHEmptyRecycleBin

The function exists "SHEmptyRecycleBinA", Some examples of how to use it in a Visual Basic program can be found at:

http://support.microsoft.com/support/kb/articles/Q164/7/87.ASP http://www.TheScarms.com/vbasic/rundll.asp

I would assume that it can be done, but getting the right parameters and using them correctly is the key. I call api functions to open certain dialogs or perform other functions with AutoIt. So more than likely emptying the recycle bin through an api call is possible. I found an interesting website with a ton of api functions along with the parameters.

http://www.vbapi.com/ref/funca.html

For the SHEmptyRecycleBin you need at least Shell32 V.4.71, but I have the 4.00 version and I can empty the recycle bin, so I think there is a way to do this, if you create a folder and name it: Recycle Bin {645FF040-5081-101B-9F08-00AA002F954E} You'll got another recycle bin.

.....

Unfortunately, I don't think you can do it with Rundll. The SHEmptyRecycleBin function is in the SHELL32.DLL file, but it doesn't work with RunDll, because it needs a Windows handle, unlike SHExitWindowsEx, which just takes a simple parameter.

One avenue you may want to research is Microsoft's MSDN site. I've found this to be a great resource for locating this type of thing. After a quick search I was able to find the following information: The Windows API provides the ability to perform a copy, move, rename, or delete operations on a file system object using the SHFileOperation function that is exported by Shell32.DLL.

One other option, as a developer, would be to lump the functions you need into your own library (DLL or OCX), and then call your own functions.

Q #1: What exactly does "Reveal mode" do? Does it record my keystrokes and mouse movements and if so, do I cut and paste the contents into a text file to make a script? Lets say I want to automate a software installation. Will I need to study the commands and write it in notepad or will AutoIt assist me and create a script by recording my key strokes?

A #1: The AutoIt "Reveal" mode allows you to find out the exact title and text content (if any) of any open window in order to properly specify the window within an AutoIt script (see the WinXXX commands, e.g. WinWait, WinWaitActive, WinClose, etc.)

The Reveal Mode window shows information about some other window that is active. For example certain AutoIt commands take window titles as an argument. The Reveal Mode window echoes the title of the currently active window which you can copy and paste into your script. In a similar way the other information displayed allows you to script the clicking of a button, etc.

A #2: Let me give you a scenario of how I go about creating an automation of a process. Let's say I want AutoIt to automatically bring up the desktop properties dialog box.

First, if you haven't done so already. Install AutoIt. It will make some things easier.

I would start with bringing up AutoIt in reveal mode. Reveal mode is just a way of seeing information about windows and the mouse position while you are going thru the process. All you need to do to bring up reveal mode is to click on it in the AutoIt group in the Program item on the Start menu.

Once that's done. start up an editor (NoteTab, the AutoIt Script

Editor, WinVi, etc.) and minimize it. This will allow you to write down what is being displayed in reveal mode.

Then I would start up the process I want to automate. In my example I could bring up the display dialog box in 2 different ways. But for this example I'll bring it up via the control panel.

To get to the Start menu I would normally press CNTRL-ESC. So I would put a "Send, ^{ESC}" as my 1st statement in the editor.

Next I would use the arrow keys to go up 6 menu items to the "Settings" item. So I would put "Send, {UP 6}" as the next line.

Next I would press the right arrow key to highlight the "Control Panel" item and then press <ENTER> ("Send, {RIGHT} {ENTER}").

Now I have to look at the reveal mode window and see what the title of the "Control Panel" window is. Obviously in this instance it's "Control Panel" so we need the script to wait for the window to come up so we'll put a "WinWaitActive, Control Panel" as the next line.

Lastly, we need to bring up the Display dialog box. The easiest way is to get the focus into the main part of the window by pressing an arrow key so "Send, {RIGHT}" comes next. Then in my Control Panel window I have 3 icons that start with a "D". So I would have to type 3 d's to get to the proper icon ("Send, ddd{ENTER}") and then press an <ENTER> to bring it up.

All of this together is in the script below. Of course this is a contrived example. But I'm sure you get the drift.

```
; ======== Scriptlet Starts Here ========
Send, ^{ESC}
Send, {UP 6}
Send, {RIGHT}{ENTER}
```

 Q #1: How can I schedule the running of an AutoIt script?

A #1: A strategy is to use the time variables with a compare (IfEqual) and wait loop internal to your script. placing the script into the startup folder will start it. This may eat some clock cycles but in my case the script will be only one of two applications running on the machine at the time.

A #2 If you're using Win 98 or above, you can schedule your script through the System Task Scheduler. I can't remember if Win 95 had this. Also, there are several freeWare task schedulers that have been modeled after Unix Cron.

A #3: I use Just In Time to start an AutoIt script to collect call information data every half hour from a telephone switch. The Microsoft scheduler and many others schedulers run in ring 3 and will not interrupt the kernel it a higher priority task is running. If the programmed start tick passes the MS scheduler does not start the scheduled program. Since I went to JIT I have only missed one startup because of the 56 day uptime system bug in MS 9x OS. A scheduled reboot each weekend by another script, or JIT can do it for you, JIT (Version 4.xx and below) is freeware and is available from

http://members.xoom.com/polyakoff/

Version 5.0 and up has been released as inexpensive shareware.

Here are some small scripting/scheduling/automating utilities, that may be useful for reference:

NNCron - http://nemtsev.virtualave.net/

Runner - http://home.comset.net/panoff/runner/runner.phtml

 $Control\ Agent\ -\ \underline{http://www.dg.spb.ru/cgi/index.cgi?ca/default.htm+koi8}$

Q #1: Does anyone know if an AutoIt archive exists?

A #1: An archive of the AutoIt mailing list from the listbot message board has been put into a compiled HTML file. This can be found in the files section of http://groups.yahoo.com/autoit.

The files from the AutoIt list on http://groups.yahoo.com can be found there.

A #2: "swynk.com" has started an AutoIT archive, primarily for AutoIT exposure

and help with SMS package distribution. We've managed to build a pretty strong interest over the last few weeks and folks have started sending in their scripts. But, we would be happy to start including any scripts from this list if deemed appropriate. Here's the AutoIT section:

http://www.swynk.com/trent/Sections/AutoIT.asp

SWYNK is updated bi-weekly so the script would be available for download pretty quickly. We have unlimited disk space, search engines, etc.

Q #1: Has anyone used AutoItX within a Visual Basic Application? If you have, could you send me a small sample of how to include AutoIt in a Visual Basic project?

A #1: I use AutoItX in some of my VB apps. If you are interested, I created VB Module which registers the AutoItX.dll automatically when it is used. It also makes it easy to reference the AutoItX control in your code (I have only tested it on NT 4).

Some of the functions in the Module are not AutoItX related. I also added functions for REG.EXE, SRVINFO.EXE, and NET.EXE to read and write to remote registry, get server info, and manipulate NT User Accounts. Link to the VB Module at:

http://chameleonc.com/tfiles/autoitbas.zip

Here is an application which utilizes the Module. It's a cool app. It does NT User Account maintenance and reporting. It only works on NT 4.

http://chameleonc.com/tfiles/nttools.zip

Q #1: Is there a sure way to command my Windows to reboot?

A #1: Have you tried the AutoIt shutdown command?

from the manual:

Shutdown,

This command allows you to perform various forms of shutdown. The type is determined by . The flag can be a combination from the table below:

| Function | Flag |
|------------------------------|--------|
| Tog off the assument soon | 0 |
| Log off the current user | U |
| Shutdown the workstation | n 1 |
| Reboot the workstation | 2 |
| Force closing of application | ations |
| (may lose unsaved wo | ork) 4 |
| Shutdown and power off | |
| (if supported) | 8 |
| | |

e.g. To shutdown and force applications to close without saving = Shutdown + Force = 1 + 4 = 5: Shutdown, 5

A #2: Here's the page where I found this...

http://www.onecomputerguy.com/desktop.htm#logoff_icon

And here's a list of commands that they show. It's been a long time since I've had Windows 9x, so I can't test this: sorry. I would suggest trying this from a command prompt, then working your way

outward toward a script. Isolate whether the problem is with the script or the command.

For Shutdown, the command is C:\WINDOWS\RUNDLL.EXE user.exe,exitwindows

For Restart, the command is C:\WINDOWS\RUNDLL.EXE user.exe,exitwindowsexec

For Logoff, the command is C:\WINDOWS\RUNDLL.EXE shell32.dll,SHExitWindowsEx0

A #3: With Windows98, there are several known shutdown bugs. I don't know specifically of shutdown bugs on Windows95, but you can get this same sort of problem with bad software interactions. The way to test this is to first run a normal shutdown through the Start Menu shutdown option. If that works, use the standard command "rundll32.exe user,exitwindows" in the Start Menu run option. If either one of these fails, it is probably a problem with your OS. If they both work, it's harder to say what the problem is, but knowing that this works would help someone point you closer to the right direction.

A #4: To remotely reboot you might use the "Shutgui" utility from the NT-Server Resource Kit. The NT res kit has a "Shutdown" manager that allows you to SD a PC on your network, but there is a command line 1 "SHUTDOWN" that can be run as an "AT" script on your server.

Check out the help file for both of the above for the syntax... It should be ..

SHUTDOWN \\"Computername" ..

Or see the following for more details

| http://download.mycomputer.com/detail/57/56.html |
|--|
| A #5: To currently log off the current user from WIN NT4.0, NOT 95 etc from the command line or batch script the command is this, |
| rundll32 user32.dll,ExitWindowsEx |
| Using this method removes dependencies on resource kit utils etc. what about the good old dos (use ipconfig /release // /renew) |
| Q #2: What do I do if other users are connected to the system as well? |
| A #1: Try: |
| Runwait, %COMSPEC% /c net.exe STOP |
| Q #3: I'm trying to use Shutdown in a script to create an executable to reboot a PC. The PC is on a W95 network, and accessed by others, so I get the Sharing Message Box before shutdown, telling me "there are 2 users connected to your PC, etc" and prompting for Y/N to continue with the reboot. |
| The gist of the script is: |
| Shutdown, 2; to reboot the PC (I've also tried Shutdown, 6) |
| WinWaitActive, Sharing |

; waits for the Sharing message box

Send, y

; answers Yes, to continue with reboot

It seems a simple enough script, but it never gets past the Sharing box. Has it shut down by then to a state where Autoit executables can't continue (the icon is still in the system tray, so I presume it's still running), or have I got something more fundamental wrong? If so, any solutions, anyone?

A #1: You may want to put the line that sends a !Y to the Sharing dialog box to a [ADLIB] section just in case there are no other users connected to the machine.because with the WinWaitAcive statement the script will stop running until that window appears and if no one else is attached to the box then the windows will not appear and the script will wait. Also you can try a IfWinExist statement on that line instead of the [ADLIB] section.

A #2: There's a not-normally-installed Windows95 accessory component, Net Watcher, on the W95 CD that allows you to disconnect users. I'll do a script for it on Monday to clear the connections prior to rebooting, then I can't see why Restart shouldn't work. I can't find it on W98 at home though - anyone know if it's been replaced by something different in W98?

Actually, AutoIt quits straight after a shutdown command, if it's still in the taskbar after this command it's because there is a reboot in progress.... So, once the shutdown command has been issued, no other commands (ADLIB or otherwise) will be executed.

| There's probably a way to stop all network connections before running this (on 9x and NT). Anyone seen such a program? |
|--|
| A #3: For a Win Reboot or (current) User Logoff/Logon |
| Create a shortcut to the Rundll.exe or the Rundll32.exe file and type the following Path property, eg the SYNTAX : |
| C:\Windows\RUNDLL.EXE user,exitwindowsexec |
| -one single space between .EXE and user -one single comma between user and exitwindowsexec |
| single/doubleclick the link, |
| A #4: The purpose is to release a client DHCP IP lease just before system shutdown so that a geographically roaming notebook user will automatically be serviced with a new IP address when they connect to a different DHCP server. I assume it would look something like this. ; ======= Script Starts Here ======= @ECHO OFF IPCONFIG /RELEASE Rundll32, user, ExitWindowsExec (I know this syntax is incorrect, ideas?); ======= Script Ends Here ======= |
| A #5: For those interested in techie stuff |

NT is very easy to shutdown from the API and it also has the facility to "force" shutdowns (i.e. any open programs and data are dumped) - This is why the code in AutoIt will always work under NT.

Different story under 95, internally I'm using the ExitWindows API but under 95 there is NO way to force a shutdown with the API (bit of a bonkers decision on MS's part in my opinion). So each shutdown utility for 95 will work in a different way depending on how the author tried to force the issue. I'm using some code from MSDN which basically gets a list of all processes running and tries to kill them. If they haven't "died" within a couple of seconds ("Please save data - blah blah blah") then they are given a kick and really killed. After this the normal windows "ExitWindows" API is used - which usually succeeds if all applications have been closed.

This works for most 95 machines but I've also seen the "Please wait for windows to shutdown" message as well (and I've also seen that message quite a few times on manual shutdown!)

Anyway, that's just for info, and that's why there are soooooo many 95 shutdown applications that don't work 100%.

Q #4: I would like to know how to run a script on a local network from a Windows 98 "server". For exemple, i would like to shutdown all computers of my local network in the same way. is it possible?

A #1: You could make the script and then compile it. Place it on the network and then have a link to it for the users. I also email the compiled scripts as attachments when convenient. This allows the receipient to run the script immediately upon getting the email.

| Q #1: Is it possible to send keystrokes to invisible windows with AutoIt? |
|---|
| A #1: No |
| Q #2: Is their anyway to have AutoIt hold a mouse click down (for a second or so)? The reason I ask is some Applications (Adobe Acrobat in this case) have "flyout menus" that are activated by holding the mouse down over a button for a second or so. Is their anyway to do this with AutoIt? |
| A #1: I haven't tested it yet, but you might try the SetKeyDelay command before the LeftClick command. I've had luck doing this with LeftClickDrag command. Syntax example from my script follows. ; ====== Scriptlet Starts Here ====== SetKeyDelay, 400 LeftClickDrag, 590, 235, 247, 235 SetKeyDelay, 20 ; ====== Scriptlet Ends Here ======= |
| |

A #2: I have not done this, but I believe if you run AutoIT's Reveal Mode utility, and then mimic the mouse movements yourself, the utility will identify the specific coordinates relative to the display. Then, try using the MouseMove function to initially position the mouse at the appropriate location, and then use the LeftClickDrag function, specifying the same coordinates in both the x1/y1 and x2/y2 variables. You might also try using the Repeat/EndRepeat functions to extend the left-click operation to the desired length.

Q #3: Its kind of annoying having the mouse position fly off to some odd position after the various mouse operations (LeftClick etc.). Is it possible to bring it back to where it was originally? A #1: yes you can do it with: ; ====== Scriptlet Starts Here ====== WinActivate, Program Manager MouseGetPos, X, Y WinActivate, [your application] [Mouse Operation: LeftClick etc ...] WinActivate, Program Manager MouseMove, %X%, %Y% WinActivate, [your (new) application window] ; ====== Scriptlet Ends Here ====== But it's somewhat slow, messy and problematical, having to activate the appropriate application window at the end etc. Q #4: Does anybody know a clean way to determine that Windows has finished booting. I want an autoit script to run after all programs from the windows startup have finished, otherwise some windows popping up during the boot process interfere with my script. Starting the script with a long Sleep is not very 'clean'.

| A #1: Maybe trying WinWait for all of the programs that are supposed to be loading; if it hasn't loaded, wait until it does. Do that for all of the programs. |
|--|
| A #2: I'm not sure about the sequence of loading of windows programs, and I know this is a bit artificial, but could you put notepad.exe in the startup group and just winwait till it is activeclose it and start your script? |
| A #3: I don't think there is any way to do this. There are so many programs that can start at bootup, and they can start from so many different places, that I don't think there will ever be a way to do this. I know it's a kluge, but I would: ; ====== Scriptlet Starts Here ====== repeat, 10 sleep, 30000 endrepeat Winminizeall; ====== Scriptlet Ends Here ====== |
| A #4: Piece of cake :) WinWait, Program Manager |
| Q #5: How do I create a shortcut from within AutoIt? |

A #1: You have to use the Shortcut Wizard that's part of the Desktop Explorer shell (Program Manager). The following script should work on all Windows versions and resolutions, but no guarantees.

; ====== Script Starts Here ====== ; Get the screen resolution. GoSub, FindResolution EnvDiv, Horizontal, 2 EnvDiv, Vertical, 2

; Bring up a shortcut creation wizard. WinActivate, Program Manager RightClick, %Horizontal%, %Vertical% Send, ws

; Specify the full UNC. WinWaitActive, Create Shortcut Send, C:\\autoexec.bat{ENTER}

; Specify the shortcut text. WinWaitActive, Select a Title for the Program Send, Test 1{ENTER}

WinMinimizeAllUndo Exit

FindResolution:

; Find out what resolution the display is at.

; Find out what OS we're on.
IfEqual,A_OSTYPE,WIN32_WINDOWS, goto, Win9x
IfEqual,A_OSTYPE,WIN32_NT, goto, WinNT

Win9x:

; Get the resolution from the registry.

RegRead, VideoDevice, REG_SZ, HKEY_CURRENT_CONFIG,
Display\\Settings, Resolution

; Get the Vertical resolution.
StringGetPos, P, VideoDevice, \,
StringLen, L, VideoDevice
SetEnv, L2, %P%
EnvAdd, L2, 1
StringTrimLeft, Vertical, VideoDevice, %L2%

; Get the Horizontal resolution.
SetEnv, L2, %L%
EnvSub, L2, %P%
StringTrimRight, Horizontal, VideoDevice, %L2%
Return

WinNT:

; Get the resolution from the registry.

regread, Videodevice, REG_SZ, HKEY_LOCAL_MACHINE,
HARDWARE\\DEVICEMAP\\VIDEO,
\\Device\\Video0

StringtrimLeft, Videodevice1, videodevice, 48 StringtrimRight, videodevice2, videodevice1, 8

; Put the proper values in the variables.

RegRead, Horizontal, REG_DWORD, HKEY_CURRENT_CONFIG, System\\CurrentControlSet\\SERVICES\\%videodevice2%\\DEVICE DefaultSettings. XResolution

RegRead, Vertical, REG_DWORD, HKEY_CURRENT_CONFIG, System\\CurrentControlSet\\SERVICES\\%videodevice2\%\\DEVICE DefaultSettings. YResolution

Return

; ====== Script Ends Here ======

A #2: You might want to check out a program called PC Updater. It will create shortcuts, etc. You can use variables to send it exactly where you want. Works on NT, and 9x boxes. Also, it works very well with

profiles. A great program for software installs also. Makes self extracting EXE's. I use it for installs if they're too complicated for AutoIt.

A #3: Here is a command line utility I wrote to create a shortcut. You can create the "group" by just creating a directory within the StartMenu tree. Attached is a console (DOS window) program that will take a text file and create a shortcut from it. Also attached iis the Visual C++ source for the program.

Source File Format

Line 1 - Target Link File

Line 2 - Source Executable File

Line 3 - Description

Line 4 - Command Line Args

Line 5 - Starting Directory

Line 6 - Icon File

Line 7 - Icon Index

A #4: make the shortcuts ready to work in the directory from where you install the program after the installation is done copy it with lnkcopy in the users profile.

A #5: Why not create a shortcut that has a filename of C:\autoexec.bat in the "Target" location, and then at the end of the installation when you know what the path is you modify the shortcut to add the real path and filename into the "Target" and "Start in" locations?

Or do what most installation programs do. Wait until the installation is complete, and then make the shortcut when you know

| the path and filename. | |
|---|---|
| A #6: this is where you can download the scut.exe, | |
| http://www.jsiinc.com/TIP0400/rh0422.htm | |
| This program has all the capabilities of the full Microsoft shortcut.exe program except the ability to create them. | |
| A #7: Have you thought about using a .pif file for the shortcut? | |
| I don't use AutoIt to create my shortcuts - since I wrap my AutoIT scripts in either an SMS Installer executable or a Wise InstallMaster Executable. | [|
| Q #6: In a script I want to detect a window to be active. The problem is that the windowtext varies. The text is: "A\xxxxxxxxxxX\B", where and B are constant and xxx varies. The xxx are depending on Project name and path. | 4 |
| How can I solve this? | |
| A #1: I think you can use StringRight and StringLeft to get the first two and the last two characters from the title into variables and then test IFEqual to A\\ and \\B. | |
| | |

A #2: Copy the Window title into a variable, use the String functions to

| parse it for t | the proper text? | |
|----------------|------------------|------|
| | | |

A #3: Try to use the command SetTitleMatchMode.

Q #1: Is there an easy way to change the Gateway IP (in TCP/IP) for a network card using AutoIt?

A #1: If you must automate this, I would suggest using the built in tool netsh, or the resource kit tool netset.

For example try this at the command line:

netsh interface ip set address name="Local Area Connection" gateway=10.10.123.2

For AutoIt, try below: Note that 010010123002 corresponds to 10.10.123.2

; ====== Scriptlet Starts Here ======

send, ^{ESC}sn

Winwaitactive, Network and Dial-up Connnections

send, l{ENTER}

Winwaitactive, Local Area Connection Status

Sleep, 250

Send, !p

Winwaitactive, Local Area Connection Properties

Sleep, 500

Send, i!r

Winwaitactive, Internet Protocol (TCP/IP) Properties

Sleep, 250

Send, !s!d010010123002{ENTER}

Winwaitactive, Local Area Connection Properties

Sleep, 250

Send, !w!w{TAB}{ENTER}

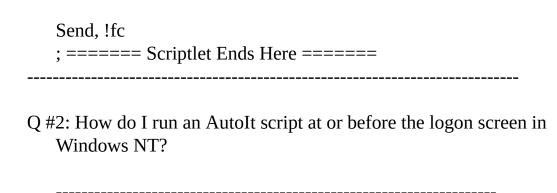
Winactivate, Local Area Connection Status

Winwaitactive, Local Area Connection Status

Send, !c

Winactivate, Network and Dial-up Connnections

Winwaitactive, Network and Dial-up Connnections



A #1: Using the RunServices key in the registry will allow you to have an AutoIt program run before login.

Microsoft has a Knowledge Base article on the various Run keys in the registry, and what order they are executed in the startup process:

http://support.microsoft.com/support/kb/articles/Q179/3/65.ASP

Here is some of it:

Under Windows 9x, where all keys are supported, the keys are loaded in the following order:

HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\ RunServicesOnce

HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run\$ <Logon Prompt>

 $HKEY_LOCAL_MACHINE \\ \ Software \\ \ Microsoft \\ \ Windows \\ \ Current \\ \ Version \\ \ Run \\ \ Current \\ \ Version \\ \ Run \\ \ Current \\ \ Version \\ \ Run \\ \ Current \\ \ Version \\ \ Run \\ \ Current \\ \ Version \\ \ Run \\ \ Current \\ \ Version \\ \ Run \\ \ Current \\ \ Version \\ \ Run \\ \ Current \\ \ Version \\ \ Run \\ \ Current \\ \ Version \\ \ Run \\ \ Current \\ \ Version \\ \ Run \\ \ Current \\ \ Version \\ \ Run \\ \ Current \\ \ Version \\ \ Run \\ \ Current \\ \ Version \\ \ Run \\ \ Current \\ \ Version \\ \ Run \\ \ Current \\ \ Version \\ \ Run \\ \ Current \\ \ Version \\ \ Run \\ \ Current \\ \ Version \\ \ Run \\ \ Current \\ \ Version \\ \ Run \\ \ Current \\ \ Version \\ \ Run \\ \ Current \\ \ Curren$

 $HKEY_LOCAL_MACHINE \\ \label{local} Software \\ \label{local} Windows \\ \label{local} Current \\ \label{local} Version \\ \label{local} Rundle \\ \label{local} Windows \\ \label{local} Current \\ \label{local} Version \\ \label{local} Rundle \\ \label{local} Windows \\ \label{local} Current \\ \label{local} Version \\ \label{local} Rundle \\ \label{local} Windows \\ \label{local} Current \\ \label{local} Version \\ \label{local} Rundle \\ \label{local} Windows \\ \label{local} Version \\ \label{local} Windows \\ \label{local} Version \\ \label{local} Version \\ \label{local} Version \\ \label{local} Windows \\ \label{local} Version \\ \label{local} Version \\ \label{local} Version \\ \label{local} Windows \\ \label{local} Version \\ \label{local} Windows \\ \label{local} Version \\ \label{local} Version \\ \label{local} Version \\ \label{local} Windows \\ \label{local} Windows \\ \label{local} Version \\ \label{local} Version \\ \label{local} Windows \\ \label{local} Version \\ \label{local} Version \\ \label{local} Version \\ \label{local} Version \\ \label{local} Windows \\ \label{local} Version \\ \label{loc$

 $HKEY_CURRENT_USER\Software\Microsoft\Windows\Current\Version\Run$ $StartUp\ Folder$

HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOr.

With the exception of the HKEY_LOCAL_MACHINE\...\RunOnce key, all keys and their entries are loaded asynchronously. Therefore, all entries in the RunServices and RunServicesOnce keys can potentially run at the same time.

Entries in the HKEY_LOCAL_MACHINE\...\RunOnce key are loaded synchronously in an undefined order.

Because the HKEY_LOCAL_MACHINE\...\RunOnce key is loaded synchronously, all of its entries must finish loading before the HKEY_LOCAL_MACHINE\...\Run, HKEY_CURRENT_USER\...\Run, HKEY_CURRENT_USER\...\RunOnce, and Startup Folder entries can be loaded.

The RunServicesOnce and RunServices keys are loaded before the user logs into Windows 95. Because these two keys run asynchronously with the Logon dialog box, they can continue to run after the user has logged on. However, since HKEY_LOCAL_MACHINE\...\RunOnce must load

synchronously, its entries will not begin loading until after the RunServicesOnce and RunServices keys have finished loading.

Because of different system configurations (such as a computer that is configured to automatically log on), any application that is dependant upon other applications that are executed under these keys having completed must be prepared to wait until these applications are complete. Other than this exception, the above description applies to Windows NT 4.0 and Windows 2000.

One thing to remember about NT/2K. Even if you have an autoit script running, it cannot do the CTRL+ALT+DEL needed to get to the logon dialog.

another point

If you want to run something in the HKEY_LOCAL_MACHINE...RunOnce to run asynchronously you can use the command:

%COMSPEC% /C START <path\file.exe>

This works because the RunOnce executes the %COMSPEC% command synchronously with the other RunOnce commands, but the START creates a new process that is not bound by the RunOnce constraints.

Q #3. Can I use AutoIt to login a user on Windows NT or 2000

A #1: No. AutoIt cannot send the CTRL+ALT+DEL key sequence required to bring up the logon dialog box in NT/2000. This is not a limitation of AutoIt.

However, in Windows 2000 you can set the machines to not require CTRL-ALT-DEL. Go to "Users and Passwords" and disable it. But this 'auto-logon' feature will not work if a third pary utility is installed (for instance Novell's Novell Client), which takes login precendence over Microsoft's) and thus the DefaultDomain, DefaultLogin, DefaultPassword entries in the Registry won't work.

From the AutoIt documentation:

"Windows does not allow the simulation of the "CTRL-ALT-DEL"

combination!"

Q #1: How do you save an uninstall string value in a variable, and run it? For example

;Check's for the uninstall string of Acrobat 3.01 RegRead, Acrobat3, REG_DWORD, HKEY_LOCAL_MACHINE, Software\\Microsoft\\Windows\\CurrentVersion\\Uninstall\\Adobe Acrobat 3.01, UninstallString

;Runs the Uninstall for Acrobat 3.01 Run, Acrobat3

I tried to run with the variable Acrobat3 and %Acrobat3 but none work.

A #1: I think the type of the uninstall entry is REG_SZ

Q #2: How can I read/write default entries to/from the registry?

A #1: To read/write a default entry from/to the registry use the RegRead or RegWrite command and fill in a blank, where the ValueName is required. The following is wrong because the word "(Default)" is specified at the end. Use an empty string ("") or simply nothing at all instead of (Default).

RegRead, version, REG_SZ, HKEY_LOCAL_MACHINE, SOFTWARE\\Classes\\Classes\\BureauVanDijk\\Promedia\\Version, (Default)

The line should be:

RegRead, version, REG_SZ, HKEY_LOCAL_MACHINE, SOFTWARE\\Classes\\Classes\\BureauVanDijk\\Promedia\\Version,

- OR -

RegRead, version, REG_SZ, HKEY_LOCAL_MACHINE, SOFTWARE\\Classes\\Classes\\BureauVanDijk\\Promedia\\Version,

Q #3: How do I delete a registry key?

A #1: To delete a registry Value, use the AutoIt command: RegDelete. To delete a registry Key, use reg.exe from the resource kit. For example, mine is in d:\2kreskit:

Run, D:\\2kreskit\\reg.exe delete HKLM\\Software\\Microsoft\\Windows\\CurrentVersion\\Explorer

(BTW, please don't execute the above line, it's just an example, the 'Explorer' key is pretty useful to keep around)

The command syntax is:

C:\>reg delete /?

Registry Console Tool For Windows 2000 - version 2.0 Copyright (C) Microsoft Corp. 1981-1999. All rights reserved

REG DELETE KeyName [/v ValueName | /ve | /va] [/f]

KeyName [\\Machine\]FullKey
Machine Name of remote machine - omitting defaults to the current machine
Only HKLM and HKU are available on remote machines
FullKey ROOTKEY\SubKey

ROOTKEY [HKLM | HKCU | HKCR | HKU | HKCC]
SubKey The full name of a registry key under the selected
ROOTKEY

ValueName The value name, under the selected Key, to delete When omitted, all subkeys and values under the Key are deleted

/ve delete the value of empty value name <no name>

/va delete all values under this key

/f Forces the deletion without propmt

Examples:

REG DELETE HKLM\Software\MyCo\MyApp\Timeout
Deletes the registry key Timeout and all of its subkeys and values

Note: With reg.exe you can delete registry keys remotely. The command-line syntax is:

reg delete [ROOTKEY\]Key[\Value] [\\Machine]

Parameters in square brackets are optional

REG DELETE \\ZODIAC\HKLM\Software\MyCo /v MTU Deletes the registry value MTU under MyCo on ZODIAC

A #2: this isn't possible without calling an external program from AutoIt. i.e. use Kixtart,

```
; ====== Kixtart Scriptlet Starts Here ======
...
delkey("HKEY_LOCAL_MACHINE\Software\Something")
...
; ====== Kixtart Scriptlet Ends Here ======
and then call the kixtart script from within AutoIt like this:
```

| runwait,%COMSPEC% /c kix32.exe mykixscript.kix,, |
|---|
| Q #4: I need a way that doesn't need an external executable, What can I write in a .REG file |
| A #1: I remember that I read that this works: |
| ===== Begin .REG-File ===== |
| REGEDIT4 |
| [-HKEY_LOCAL_MACHINE\Software\Something] |
| ===== End .REG-File ====== |
| then call |
| runwait, %COMSPEC% /c regedit /s MyRegFile.reg,, |
| The "-" symbol in front of the key is what tels regedit to delete the key. |
| Q #5: Is there a way from AutoIt to secure a Registry key? |
| |

A #1: there's no built-in command in AutoIt, but you can do this with the command line utility regini.exe from the NT Resource Kit. At:

http://www.jsiinc.com/subb/tip0700/rh0723.htm

The following article can be found.

0723 A better way to set registry permissions on a key and its' sub-keys, in batch. Supplement Two of the NT 4.0 Server Resource Kit contains Regini which will allow you to set registry permissions in the HKEY_LOCAL_MACHINE and HKEY_USERS hives. Regini reads a script

file, which for our purposes, contains a single record in the \Registry\Key [ACL] syntax:

\Registry - A constant.

Key:

\Machine = HKEY_LOCAL_MACHINE \User = HKEY_USERS

[ACL] - The ACL is a list of numbers, separated by spaces, within square brackets. The numbers represent the following permissions:

- 1. Administrator Full
- 2. Administrator R
- 3. Administrator RW
- 4. Administrator RWD
- 5. Creator Full
- 6. Creator RW
- 7. World Full
- 8. World R
- 9. World RW
- 10. World RWD
- 11. Power Users Full
- 12. Power Users RW
- 13. Power Users RWD
- 14. System OpFull
- 15. System OpRW
- 16. System OpRWD
- 17. System Full
- 18. System RW

- 19. System R
- 20. Administrator RWX

Examples:

\Registry\Machine\System\CurrentControlSet\ENUM [1 8 17] will grant:

- 1 Administrator Full Control,
- 8 Everyone Read, and
- 17 System Full Control.

\Registry\User\S-1-5-21-2053067395-480382929-641664369-1001\\ Software\Strange Software Thingy [1 8 17] - Same as above.

\Registry\Machine\System\CurrentControlSet\ENUM [8 17] - Will remove the Administrator group from the first example.

REG (from the Resource Kit Supplement, not the freeware at tip 170) has a Query option that will allow you to display keys and sub-keys. The JSIRegS.bat file will apply the specified ACL to the Key and its' sub-keys. The usage is:

JSIRegS ScriptFile1 [ScriptFile2 ScriptFile3 . . . ScriptFileN]

JSIRegS.bat contains:

@echo off
:check
PushD <Path to Resource Kit>
if exist %TEMP%\JSIRegS.log del /q %TEMP%\JSIRegS.log
if exist %TEMP%\JSIRegS.ini del /g %TEMP%\JSIRegS.ini

if exist %TEMP%\JSIRegS.ini del /q %TEMP%\JSIRegS.ini

if /i "%1"=="" goto syntax

if exist %1 goto parse

:syntax

@echo INI file %1 not found or contains an invalid entry.

goto end

:hku

set reg=%reg:\Registry\User=HKU%

```
set reg=%reg:\registry\user=HKU%
   if /i "%reg%"=="%savekey%" goto Ssyntax
   goto key
   :parse
   for /f "Tokens=1-2* Delims=[]" %%i in (%1) do set reg=%%i!&set
sec=%%j
   set reg=%reg: !=%
   set savekey=%reg%
   set reg=%reg:\Registry\Machine=HKLM%
   set reg=%reg:\registry\machine=HKLM%
   if /i "%reg%"=="%savekey%" goto hku
   :key
   regini %1
   set reg="%reg%"
   reg query %reg% /s >> %TEMP%\JSIRegS.log
   for /f "Tokens=*" %%i in ('findstr /b "[" %TEMP%\JSIRegS.log') do call
:mkini "%%i"
   POPD
   set reg=
   set savekey=
   set sec=
   set sub=
   if exist %TEMP%\JSIRegS.log del /q %TEMP%\JSIRegS.log
   SHIFT
   if "%1"=="" goto end
   goto chck
   :mkini
   set sub=%1
   set sub=%sub:[=%
   set sub=%sub:]=%
   set sub=%sub:"=%
   @echo %savekey%\%sub% [%sec%] > %TEMP%\JSIRegS.ini
   regini %TEMP%\JSIRegS.ini
   del %TEMP%\JSIRegS.ini
   :end
```

Q #1: How do I run an external command like MS Word?

A #1: Try:

Run, C:\\Program Files\\Microsoft Office\\Office\\Winword.exe

Or

Run, %COMSPEC% /c C:\\Program Files\\Microsoft Office\\Office\\Winword.exe

Or

Send, ^{ESC}Rwinword{ENTER}

The Run command can be substituted by RunWait. The only difference is that RunWait suspends the AutoIt script until the DOS command has finished, and the Run command allows the AutoIt script to continue at the same time as the DOS command.

Q #2: Is there a way to activate the Run box (Winkey-R) dialogue quickly (to execute the contents of the clipboard). Anyone know of some kind of Rundll32 sequence that will do this (activate the Run dialogue box) or other programs that will do this?

I know the Run dialogue box on the start menu and can Winkey to it that way but it's kind of iffy especially if you have other programs starting with R.

I've tried the "Run %Clipboard%" etc., but nothing is as effective as the Run dialogue box for figuring out the syntax of various commands and understanding spaces etc. I think I've tried just about everything for executing text (specifically for "executing the clipboard text contents"). Nothing works as effectively for executing a wide variety of text commands as that Run dialogue box. As an example let me illustrate as the following 2 lines of text:

Notepad "C:\File with spaces in it.txt" C:\File with spaces in it.txt

That Run dialogue box is the only thing I know of that will run either line without problems. Using your suggestion it seems I'd be doing something like:

SetEnv, variable, %COMSPEC% /c %CLIPBOARD% Run, %variable%

Or possibly more simply: Run, %COMSPEC% /c %CLIPBOARD%

This will work for the first line:
Notepad "C:\File with spaces in it.txt"

But not the second: C:\File with spaces in it.txt

If one tries to add the extra quotes for the second line, then the first line won't work.

A #1: If you want to do Winkey+R, try

send, {LWIN}R

{LWIN}R = Left Windows key + R (although it isn't the same as actually hitting the winkey and R together)

A #2: What about putting the application to be executed in an .INI file, which can have variables specified how you want them (but again they must follow the specific criteria you've already built into your script (i.e. Are they going to have quotes or not, Will they call the program and the file, or let the OS judge which program to open the file, etc...)

If the variable of %CLIPBOARD% itself is such an uncontrolable factor (possibly entered by the user, which it may or may not have Quotes, or may or may not have the application followed by the filename) Then perhaps you should put limitations into the variable which reject any input given by the user that does not have quotes around it, or list some examples for the user so they know what format is accepted, and what is not.

NOTE: Another very interesting thing I've found, is that IF the variable you want executed is in an .INI file (or any text file for that matter), the variable DOES NOT need to have the obligatory '\' preceeding any special character. the script reads the variable from the .INI file or equiv., then that variable then becomes....'converted' to Auto-It accepted-lingo.

try this;

setenv, variable, %COMSPEC% /c

c:\\program files\\mydirectory\\myfile.exe \/switch1 \/switch2 run, %\variable%

i.e.

instead of; you can have this;
-----\\\\server\\share\\file.exe \\Server\\share\file.exe

c:\\directory\\file.exe C:\\directory\\file.exe
%comspec% \/c c:\\directory\\file.exe %comspec% /c
c:\\directory\\file.exe

etc...

If anyone finds any flaws in this, please let me know... I have a variable path depending on which system I plan to run scripts from, and it tries first one of the paths. If it finds it it executes it, if not it tries the next path/file location (as specificed in the .INI file) in sequence.

Q #3: How do I play a WAV file from within AutoIt?

A #1: use:

Run, C:\windows\media\logoff.wav

and the WAV file will "hijack" whatever sound player you use to play the sound. I know there is a system call for the sounds but I don't know what it is.

A #2: This is the line that windows uses to play a wav file.

"C:\Program Files\Windows Media Player\mplayer2.exe" /Play "%L"

The way to get this information is to:

- * open any folder in windows explorer.
- * Click on the view menu and choose options,
- * click the file types tab
- * scroll down to the file type you want (in this case .wav).
- * Click on the edit button

* Click play and edit again.

Now you will see the application used to perform this action and in that window you will see the above line.

replace "%L" with "c:\windows\media\hi.wav" or whatever your wav file is.

Q #4: Why doesn't the DOS command window close when I do this?

Run, "net" "use r: \\\\hotfurn1\\bin" doesn't work

A #1: Try using:

Run, %COMSPEC% /c net use r: \\\\hotfurn1\\bin

Using "%comspec% /C" has the advantage of closing the window after the command finishes instead of leaving it open.

Q #5: Using the script below, how can I make sure that the default browser opens the hotmail site? I would like to have this just for the default browser. I took a look at the Open command for HTMLFile in the registry and it had this: {Registry key:

HKEY_CLASSES_ROOT\htmlfile\shell\open\command}
"C:\PROGRA~1\Plus!\MICROS~1\iexplore.exe" -nohome

{Registry key:

HKEY_CLASSES_ROOT\http\shell\open\command}
"C:\PROGRA~1\Plus!\MICROS~1\iexplore.exe" -nohome

How would I strip the -nohome from the variable? Are there any string

manipulation functions in AutoIt?

```
Using this line does not work with either browser as default...
  Run, http://www.hotmail.com,,
  ; ====== Script Starts Here ======
           -----
  ; Logs into a Hotmail account
  ; ========={ES
Get the path to Netscape from the registry
  ;RegRead, NSPath, REG_SZ, HKEY_LOCAL_MACHINE,
      SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\App
   Paths\\Netscape.exe,
   ;Run,%NSPath% http://www.hotmail.com,,
  ; Get the path to IE from the registry
  11/01/00}==============
  ;RegRead, IEPath, REG_SZ, HKEY_LOCAL_MACHINE,
    SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\App
Paths\\IEXPLORE.EXE,
  ;Run,%IEPath% http://www.hotmail.com,,
  WinWaitActive, Hotmail
  : Wait 5 seconds
  Sleep, 5000
```

```
; No longer needed as Hotmail now uses javascript to
  ; move the cursor to the UserID field
  ; You may have to change these coordinates depending upon your screen
resolution
  ;LeftClick, 260, 210
  ; Replace xxxxxxx with your username:
  Send, xxxxxxxx
  Send, {TAB}
  ; =========={{ES
; Extra tab is now needed
  11/01/00}============
  Send, {TAB}
  ; Replace yyyyyyy with your password:
  Send, yyyyyyyy
  Send, {TAB}
  Send, {ENTER}
  ; ====== Script Ends Here ======
A #1: Try this:
```

run, %comspec% /c start http://www.hotmail.com,, min

Q #1: How do I output spaces that are at the beginning of a line? A #1: The SEND command gobbles whitespace at the beginning and end of lines. For example: sample file sample.txt: ----this is line one ----this is line two ----this is line three ********** ;--sample.aut file FileReadLine,line1,C:\\sample.txt,1 FileReadLine,line2,C:\\sample.txt,2 FileReadLine,line3,C:\\sample.txt,3 run, notepad. exe winwaitactive, Untitled Send,%line1%{ENTER} Send,%line2%{ENTER} Send,%line3%{ENTER} msgbox,0,test,%line1%\n%line2%\n%line3% ********** output in msgbox is fine output to notepad is: ----this is line one

```
----this is line two
   ----this is line three
   I fixed by doing the following:
   Send,#%line1%{ENTER}
   Send,#%line2%{ENTER}
   Send,#%line3%{ENTER}
   The pound sign does not get sent, but it does allow you to delineate the
   start of the string.
Q #2: With AutoIT can I work with excel cells,
   i.e.: can I put values inside cells
A #1: Yes. The same way you would normally do
   keystrokes and mouse clicks. An example
   might be the following scriptlet.
   ; ====== Scriptlet Starts Here ======
   ; Put EXCEL's PATH into the EXCEL variable.
   SetEnv, EXCEL, F:\\Apps\\Office97\\Office\\excel.exe
   ; Run the Excel program.
   Run, %EXCEL%
   ; Wait for Excel to be ready for input.
   WinWaitActive, Microsoft Excel
   ; Goto cells C1-C5 and enter #'s & a formula
   ; that finds 10% of the 4 cells above C5.
   Send, \Gc1{ENTER}22.5{ENTER}
   Send, \Gc2{ENTER}55.625{ENTER}
   Send, \Gc3{ENTER}12.75{ENTER}
```

Send, \Gc4{ENTER}103.31{ENTER}.
Send, \Gc5{ENTER}=sum(C1-C4)/100*10
; ====== Scriptlet Ends Here =======

| Q #1: I've got a script where I want to add network printers. It appears that I can't get the backslash "\" to be input. For example I want to type in:: |
|---|
| \\servername\.sys.vol1.servername |
| The output I get is: |
| \servername.sys.vol1.servername |
| A #1: In order to get a backslash "\", you must use a backslash in front, For two back slashes in your script, use four. For one backslash use two. Using your example, you would need to enter the line as follows; \\\\servername\\.sys.vol1.servername |
| Q #2: In relation to starting the Add Printer wizard under Windows 95. The command line is - rundll32.exe sysdm.cpl,InstallDevice_Rundll Printer but AutoIt thinks that anything after the ',' is the working directory. Anybody know a work around for this one? |
| A #1: Try putting a backslash (\) in front of the comma. |
| Q #3: Why doesn't: |
| run, "net" "use r: \\hotfurn1\bin" |

| work? | | | | |
|-------|------|------|------|---|
| | | | | |
| | | | | - |

A #1: Try using

Run, net use r: \\\hotfurn1\\bin

instead. What's been overlooked is that the backslash is an escape character in AutoIt, so you have to put it twice.

| installation process. I don't want it to be in the middle of the screen, I want to place it on the upper left corner, Is there a way to do that. |
|--|
| A #1: Display the splashtext screen, and then move it to the new location. |
| ; ====== Scriptlet Starts Here ====== SplashTextOn, 300, 300, Installation in Progress, Part 1 of 10 WinMove, Installation in Progress, , 20, 20, default, default SplashTextOn, 300, 300, Installation in Progress, Part 2 of 10 WinMove, Installation in Progress, , 20, 20, default, default SplashTextOn, 300, 300, Installation in Progress, Part 3 of 10 WinMove, Installation in Progress, , 20, 20, default, default ; ======= Scriptlet Ends Here ======= |
| etc. You get the idea. |
| Q #2: Can a picture be put in the SplashText window? |
| |

A #1: Unfortunately, no. There is presently no way to put a picture into the SplashText window.

Q #1: How do I remove spaces from a string?

I am removing spaces from telephone numbers as I am using ini files as data storage for connection (plus other) information about remote sites. While my current modem will pause for 5 secs on a space my ISDN TA "see's" a space as a delimeter i.e. the number 01234 567890 would be to dial 01234 using security protocol 567890 (which is invalid)

I have attempted to use the

```
StringReplace, Number, No, , and StringReplace, Number, No, " ",
```

function, but it won't find spaces. I have attempted nibbling the Variable (1 char at a time)

```
; ====== Scriptlet Starts Here ======
Start:
StringLeft, TST, No, 1
StringTrimLeft, No2, No, 1
SetEnv, No, %No2%
IfNotEqual, TST, , SetEnv, %Number%%TST%
Goto, Start
; ====== Scriptlet Ends Here ======
```

but I lose the whole thing then and I don't know why It should add the chars 1 by one to the Number variable unless it is a space.

A #1: Use the space trimming to your advantage! This works if you just want to remove spaces. If you want to replace them with something else it doesn't work.

```
; ====== Scriptlet Starts Here ======
; For a general removal of spaces:
```

```
SetEnv, test, this is a test A really go od test
SetEnv,Output,
loop:
StringLeft,dummy,test,1
SetEnv,Output,%output%%dummy%
StringTrimLeft,test,test,1
IfNotEqual,test,,goto,loop
MsgBox,0,test,%output%
; ======= Scriptlet Ends Here ======
```

A #2: Dump the data to a ini file and strip all the spaces with an external program such as GSAR (general search and replace (DOS)).

```
; ====== Scriptlet Starts Here ======
FileReadLine, No, numbers.txt, %Ln%
IniWrite, %No%, temp.ini, Temp, Number
RunWait, %COMSPEC% /C gsar -io "-s " -r %S%temp.ini, c:\\temp, Min
IniRead, Number, temp.ini, Temp, Number
; ====== Scriptlet Ends Here ======
```

A #3: You can do a script with KiXtart 95... All you need is a way to make the two scripts comunicate with an ini file.

First of all, you need to find the place of each space, then you begin to remove them from end to start...

you'll do something like:

```
; ====== Kixtart Scriptlet Starts Here ======

$InputStringLen = Len($InputString)

$FirstStringChar = $SpaceChar - 1

$SecondStringChar = $SpaceChar + 1

$FirstString = SubStr($InputString, 0,
```

```
$FirstStringChar)
   $SecondString = SubStr($InputString,
   $SecondStringChar, $InputStringLen)
   $CoolString = $FirstString + $SecondString
   ; ====== Kixtart Scriptlet Ends Here ======
   This way you can remove the spa
Q #2: This is the first time I use a string command. I have made this:
   ; ====== Scriptlet Starts Here ======
   WinGetActiveTitle, CLIPBOARD
   StringLeft, CLIPBOARD, PROGNAME, 10
   IfNotEqual, PROGNAME, EPLAN 5.30, Goto, exit
   ; ====== Scriptlet Ends Here ======
   I get an error on the line with stringleft. What am I doing wrong?
A #1: Change the line to
   StringLeft, PROGNAME, CLIPBOARD, 10
   The syntax is:
   StringLeft, <Output Var>, <Input Var>, <number of chars to extract>
Q #3: Is there a way to return the position of a substring within a string?
   I want to delete all text from the beginning of a line up to a colon.
```

```
A #1: Use "StringGetPos".
A #2: If you don't have version 2.6 or above, then try something like
   ; ====== Scriptlet Starts Here ======
   Seteny, test, This is a string: Test String
   stringlen, strlen, test
   seteny, count,1
   loop:
    stringmid, strtest, test, %count%, 1
     ifequal, strtest, :, goto, foundit
     envadd, count, 1
    ifgreater, count, %strlen%, goto, notfound
    goto, loop
   foundit:
    stringmid, output, test, %count%, %strlen%
    msgbox, 0, Result, The resulting string is %output%
     goto, end
   notfound:
    msgbox, 0, Error, No colon found
   end:
   ; ====== Scriptlet Ends Here ======
   note, if you don't want the colon, add one to the
   count before getting the output string
Q #4: Is there a detected difference between reading a line with a blank
   space and carriage return vs. just a carriage return?
A: Yes. For an example try:
```

```
; ====== Scriptlet Starts Here ======
Filereadline,test1,c:\\testspace.txt,1
Setenv,test2,--%test1%--
IfEqual,test2,---,Msgbox,0,Test,There is a space!
IfEqual,test2,----,Msgbox,0,Test,There is no space!
; ======= Scriptlet Ends Here =======
```

Q #1: I have a script:

```
; ====== Scriptlet Starts Here ======
setenv, Sample, AutoIt
setenv, a, Sample
stringleft, \%a\%, a, 1, 8
MsgBox, 0, AutoIt, Sample = %Sample%; a = %a%
stringleft, %a%, a, 1, 8
MsgBox, 0, AutoIt, Sample = %Sample%; a = %a%
; ======= Scriptlet Ends Here =======
```

After second stringleft content of Sample - "AutoIt" converts into "S". Why?

A #1: The syntax of Stringleft is:

StringLeft,<Output Variable>,<Input Variable>,<# of chars to extract>

so why do you have the '1' and the '8' when there is supposed to be only one number?

here is what you are actually doing

the first stringleft command is setting an environment variable '%a%' Not 'a' but a completely different variable.
Unfortunately, you cannot retrieve this variable.

In the second stringleft command:

```
stringleft, %a%, a, 1, 8
```

the %a% is converted to 'Sample' and given the first character of the variable 'a' (a=Sample, first char=s)

what you are really saying in Pseudo Basic is

```
%a%=left(a,1)
which is like saying
Sample=left("Sample",1)
which is like saying
Sample="S"
Which is what you got.
I added two lines to correct content Sample:
; ====== Scriptlet Starts Here ======
seteny, Sample, AutoIt
setenv, a, Sample
stringleft, \%a\%, a, 1, 8
MsgBox, 0, AutoIt, Sample = %Sample%; a = %a%
stringleft, %a%, a, 1, 8
MsgBox, 0, AutoIt, Sample = %Sample%; a = %a%
stringreplace, %a%, a, S, AutoIt
MsgBox, 0, AutoIt, Sample = %Sample%; a = %a%
; ====== Scriptlet Ends Here ======
but have "AutoItample"?
You are misunderstanding the commands
After the line
 stringleft, %a%, a, 1, 8
a = Sample
```

```
Sample = S
the next line:
 stringreplace, %a%, a, S, AutoIt
becomes
 stringreplace, Sample, a, S, AutoIt
which is equivalent to (in Basic)
 Sample= replace(a,"S","AutoIt")
which evaluates to
 Sample = replace("Sample","S","AutoIt")
which evaluates to
 Sample = AutoItample
Your command is saying: replace all "S" in the variable a with the
word "AutoIt" and put the result in the variable Sample.
If the last two lines are changed to:
; ====== Scriptlet Starts Here ======
setenv, Sample, AutoIt
setenv, a, Sample
stringleft, \%a\%, a, 1, 8
MsgBox, 0, AutoIt, Sample = %Sample%; a = %a%
stringleft, %a%, a, 1, 8
MsgBox, 0, AutoIt, Sample = %Sample%; a = %a%
stringreplace, %a%, %a%, S, AutoIt
MsgBox, 0, AutoIt, Sample = %Sample%; a = %a%
```

| ; ====== Scriptlet Ends Here ====== | |
|--|---|
| The script works the way it was intended. | |
| Q #2: I know it is possible to use DOS variables in Autoit by refering to them between percentage symbols, but how do you export an AutoIt variable to DOS? | |
| I've tried the following script to discover the currently logged in user: | |
| RegRead, AUsername, REG_SZ, HKEY_LOCAL_MACHINE, Network\\Logon, username run, %comspec% /c set username=%Ausername% | |
| A #1: The variable you are setting exists only within the %comspec% session in which it is created and disappears immediately when %comspec% terminates, which in this case is as soon as the SET command is executed. What you are trying to do is create the variable in the *global* environment, and the way to do that differs depending on the platform: On Win9x, you must use the WinSet utility (available on the Win9x CD-ROM). On WinNT/2000, you must use the SetX utility available in the NT or 2000 Resource Kit. You can launch both of these utilities from within AutoIt (using Run or RunWait). | |
| A #2: You can discover the currently logged-in ID by executing the command NET CONFIG in a DOS box. | f |
| A #3: In Windows NT & 2000, there is already a variable with the username | |

in it. I believe it is %username%.