WINDOWS POWERSHELL INTEGRATED SCRIPTING ENVIRONMENT 4.0

Created by http://powershellmagazine.com

PowerShellMagazine

KEYBOARD SHORTCUTS

General

Create new script	CTRL+N
Open a script	CTRL+O
New PowerShell tab	CTRL+T
Open a remote tab	CTRL+SHIFT+R
Close an open tab	Ctrl+W
Go to next PowerShell tab	CTRL+TAB
Go to previous PowerShell tab	CTRL+SHIFT+TAB

NOTE: To switch between tabs using the above sequence, Console Pane must be in focus.

PowerShell ISE help	F1
Show Command	CTRL+F1

NOTE: Remember that both commands require you to select the command in the Script or Console pane, or at least place the cursor near the command, before invoking the key sequence.

Zoom in	CTRL+ADD
Zoom out	CTRL+SUBTRACT
Invoke command history	#CTRL+SPACE
Cycle through history	#TAB
Start PowerShell.exe	CTRL+SHIFT+P

Console Pane

Go to Script Pane	CTRL+I
Cycle through command history	UP ARROW DOWN ARROW
Scroll to the output	CTRL+UP ARROW

Script Pane

Close an open script	CTRL+F4
Go to next script	CTRL+TAB
Go to previous script	CTRL+SHIFT+TAB

NOTE: The shortcuts for switching between tabs are contextual. To switch between tabs using the above sequence, Script Pane must be in focus.

Start snippets	CTRL+J
Toggle regions	CTRL+M
Find in script	CTRL+F
Find next in script	F3
Find previous in script	SHIFT+F3
Replace in script	CTRL+H
Go to line	CTRL+G
Go to match	CTRL+]

NOTE: "Go to match" edit menu option will be available only when the cursor is pointed at script block beginning/end. In other words, it must be placed at the opening or closing brace.

To upper case	CTRL+SHIFT+U
To lower case	CTRL+U
Transpose lines	ALT+SHIFT+T
Start IntelliSense	CTRL+SPACE
Go to Console Pane	CTRL+D
Show / Hide Script Pane	CTRL+R
Show Script Pane top	CTRL+1
Show Script Pane right	CTRL+2
Show Script Pane maximized	CTRL+3

NOTE: Only a subset of above Script Pane keyboard shortcuts are available, depending on the current Script Pane state.

Execution

Run a script	F5
Run only selection	F8
Run current caret line	F8
Stop execution	CTRL+BREAK CTRL+C

NOTE: Using CTRL+C for script execution termination works only when no text selected in the Script or Console Pane.

Debugging (Script Pane)

Toggle breakpoint	F9
Run/Continue	F5
Step into	F11
Step over	F10
Step out	SHIFT+F11
Display call stack	CTRL+SHIFT+D
List breakpoints	CTRL+SHIFT+L
Remove all breakpoints	CTRL+SHIFT+F9
Stop debugger	SHIFT+F5

Debugging (Console Pane)

Continue	C
Step into	S
Step over	V
Step out	O
Repeat last command	Enter
Display call stack	K
Stop debugger	Q
List the script	L
Display console debug commands	H or ?

PowerShell_ISE.exe PARAMETERS

PowerShell_ISE.exe

-File "file1.ps1, file2.ps1" [Opens file1 & file2]

-NoProfile [Does not run profile script]

-MTA [Starts ISE in MTA mode]

WINDOWS POWERSHELL INTEGRATED SCRIPTING ENVIRONMENT 4.0

Created by http://powershellmagazine.com

PowerShellMagazine

ISE SNIPPETS

Snippets are an easy way to insert chunks of re-usable or template code into a script. The snippet functions are available only in ISE.

Create a new Snippet

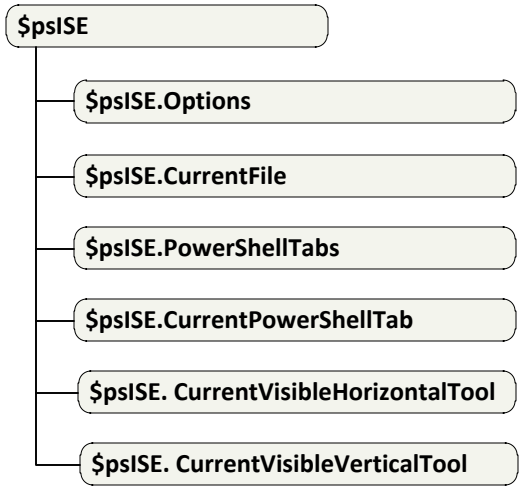
```
$textcode = 'workflow MyWorkflow{

}'
New-IseSnippet -Title "Workflow" -Text $textcode `
-Description "New workflow block"
```

Get ISE Snippets

```
Get-IseSnippet
```

ISE OBJECT MODEL

ISE exposes its underlying scripting object model to allow manipulation of various visual and functional aspects of ISE. $psISE is the root object of the ISE object hierarchy.



The **$psISE.CurrentVisibleHorizontalTool** and **$psISE.CurrentVisibleVerticalTool** objects are available only when an add-on--for example, the ShowCommands add-on--is visible in ISE.

$psISE.Options

Defines the ISE color scheme and appearance-related options. For example, use these options to set how ISE color scheme looks, how the ISE panes appear, font size, font name, and IntelliSense options.

The color scheme and appearance options are best adjusted by using commands on the Tools -> Options menu item in ISE. Here is the other important information:

To change "most recently used" count, set $psISE.Options.MruCount to desired value between 0,32.

To disable local help, set $psISE.Options.UseLocalHelp to $false.

$psISE.Options.RestoreDefaults() restores all options to ISE defaults.

$psISE.CurrentFile

Defines the properties of the current open file in ISE Script Pane such as displayname, fullpath, encoding, etc.

$psISE.CurrentFile.Editor contains the information about the script editor and the contents of the editor.

$psISE.CurrentFile.Editor.InsertText("sample") inserts specified text at the current caret position.

$psISE.CurrentFile.Editor.Clear() clears the text in the editor.

$psISE.CurrentFile.Editor.SelectCaretLine() selects the line where cursor is placed.

$psISE.CurrentPowerShellTab

Defines the properties of the current PowerShell tab and a collection of files in the tab. Also, defines the method to extend ISE add-on menu.

$psISE.CurrentPowerShellTab.Files defines a collection of open files in the tab that can be managed the same way as $psISE.CurrentFile.

$psISE.CurrentPowerShellTab.AddonsMenu contains a collection of existing add-on menus and method to create new.

To add a new add-on menu

```
$script = { $psISE.CurrentFile.Editor.SelectCaretLine() }
$psISE.CurrentPowerShellTab.AddOnsMenu.SubMenus.Add("Select _Line",$script,"Alt+L")
```

To remove an add-on menu at index 0

```
$addon = $psISE.CurrentPowerShellTab.AddOnsMenu.Submenus
$addon.Remove($addon[0])
```

$psISE.PowerShellTabs

Defines a collection of open PowerShell tabs in ISE. Each instance of PowerShell tab contains the same properties and methods as $psISE.CurrentPowerShellTab.

$psISE.PowerShellTabs.Files lists all open files in ISE across all open PowerShell tabs.

$psISE.PowerShellTabs.AddonsMenu lists all add-on menus available across all open PowerShell tabs.

$psISE events

The $psISE scripting object model provides events when a property or collection changes within ISE. These events are usually named as PropertyChanged or CollectionChanged based on the object.

For example, the following code adds an add-on menu to all newly opened PowerShell tabs:

```
Register-ObjectEvent -InputObject $psise.PowerShellTabs -EventName CollectionChanged -Action {
if ($event.SourceEventArgs.Action -eq "Add") {
 $event.Sender[1].AddOnsMenu.SubMenus.Add("Select _Line",{$psISE.CurrentFile.Editor.SelectCaretLine()},"Alt+L") } }
```