PowerShell CodeManager 8.0

1. What it is

The main idea behind PowerShell CodeManager is to make the **reuse of code** as easy and convenient as possible.
The longer you are in coding, the more likely it is, that at some point in the past you did already code the stuff you need again right now. But where was that? How did you name that script again?
CodeManager will find it. You can search **inside all of your scripts** in a matter of seconds and find what you are looking for. And the moment you find it, it's already in your clipboard, ready to be used again, no matter which editor you use, ISE, VSCode, Notepad++ or something else.
If you want to go a step further and make the whole process even more convenient and the code easier to reuse, you can create **snippets**.
CodeManager provides all the tools you need for that.
But that's not all.
CodeManager helps you documenting your scripts or exploring the code of others, by creating an **interactive function dependency graph**, of a single function or a complete script.
CodeManager can also directly **execute scripts** and lets you quickly change or add command line **parameters** and **user context.**

Snippets?
Snippets are code fragments, functions or even short scripts that you save in a special format, so that you can quickly find them and use them again in another script. Ideally, they get a description and are generalized, meaning that all things specific to a certain script have been removed.
Snippets can be used, via the "snippets"-menu entry in the context menu of ISE or pasted with CodeManager into any editor.

In this documentation, you will come across the term ***PowerGUI*** a lot and you might not know what that is. So: PowerGUI was a very good and efficient PowerShell editor from Quest, which has been acquired by Dell some years ago. Unfortunately, they ceased development of PowerGUI since. It is still in use by some people sometimes (including me) that's why CodeManager is still supporting it, but if you are not one of those people, simply ignore all references to it.

If you have a suggestion or stumbled over a bug, please send me an email to snipman@gmx.net .

Have a nice day,
Denniver Reining

2. Installation and Prerequisites

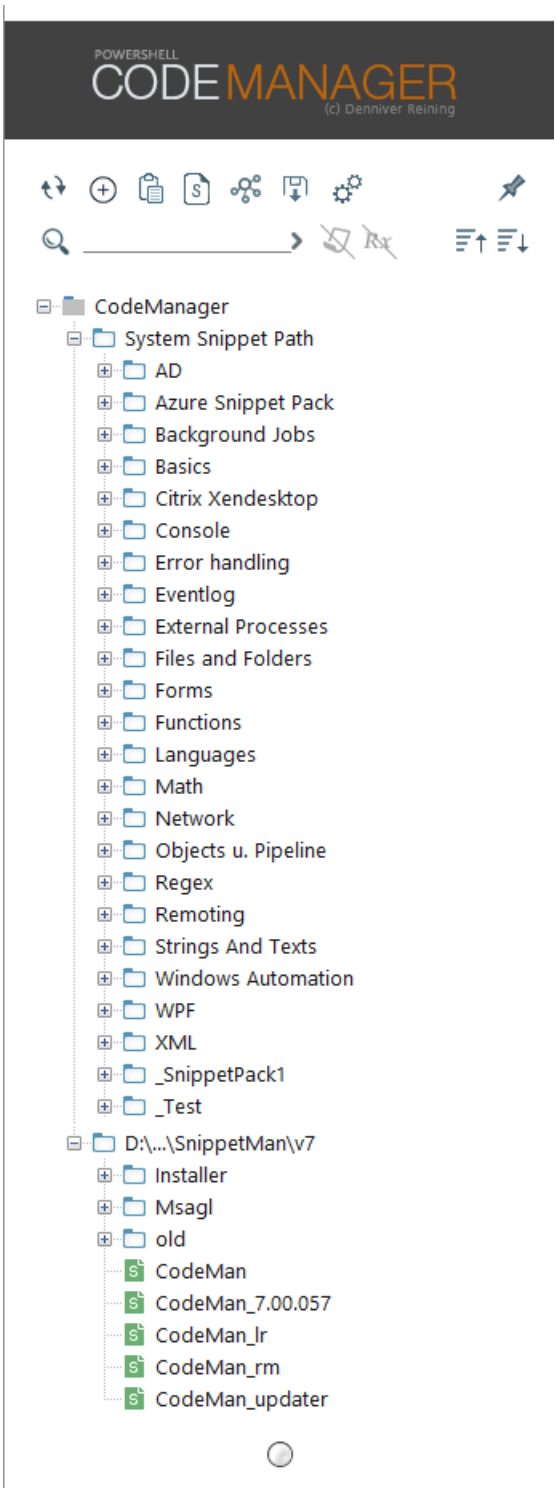
- **PowerShell V7.2** (or later) is **recommend** for this version (8.0) of CodeManager, **PowerShell V5.1** is **needed**.
- CodeManager has been tested on **Windows 10, 11, Server 2019 and 2022**, but should run fine on any other Windows version where at least PowerShell V5.1 can be installed.
- **Download** and **run** CodeManager**-**installer**.** All necessary files and folders will be created automatically. If you have any problems with the installer you can download the files in a ZIP as well.
- **Launch** CodeManager best via the shortcut (if you used the installer) or with the ***Start-xxxx.cmd*** file in CodeManager Folder.

3. CodeManager Main Window

I originally created this as an **alternative** to the insert-snippet popups in ISE (and the good old PowerGUI): They are very small, you cannot **search** for snippets there, other **locations** are not supported and ISE does not even support simple **folders**. So, if you have more than a couple of snippets (and you should), they are suboptimal to say the least. CodeManager has all the **features** I mentioned missing above (and more).

Open

- You can **open CodeManager** with the "**Floating Launcher**" (a freely moveable button on your desktop) or via the **notification icon** (near the clock)



Floating Launcher

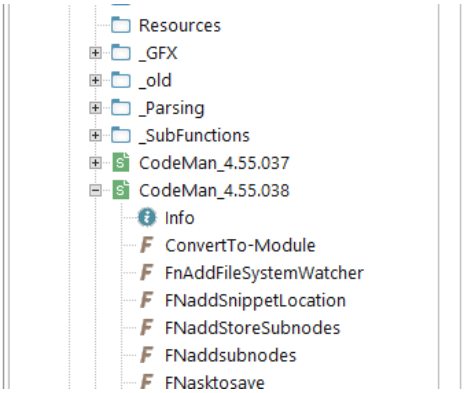
- You can **scale** the Floating Launcher in its context menu or via CodeManagers option menu and choose a **color theme**.
- To **move** the Floating Launcher around, right-click and hold.

- You can also **hide** the Floating Launcher if you like. You can then use the notification icon to launch **CodeManager**.

- ## Window Options

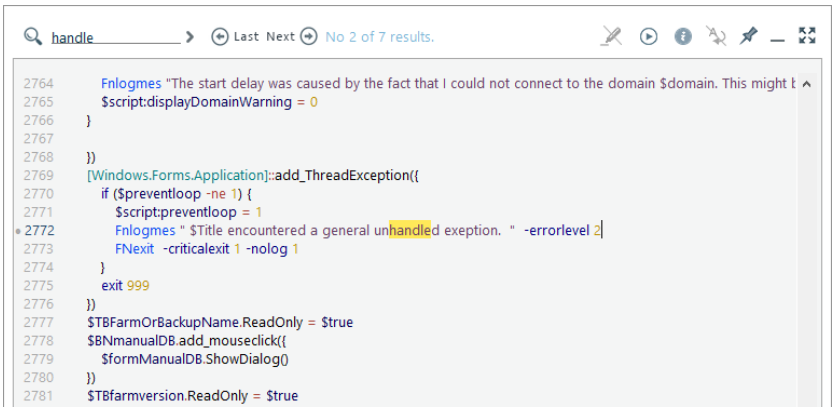
  - You can **dock** CodeManager at the right or left border of your screen (CodeManagers options menu)
  - Per default, as soon as you click somewhere outside of CodeManager window, CodeManager **hides** immediately. You can make it **stick** by clicking the "***Pin***"-button in the menu bar.
  - CodeManager has also two **large hidden blue minimize/hide buttons**, spanning top to bottom of each side of the window. And spanning the top of the window, there is a large **hidden red exit button**.

- **Script/Snippet Explorer**
  - As soon as you **single click** on a snippet or script, the code window pops up and displays its code.
  - Also, a **plus** is displayed in front of the script or snippet in the script explorer. If you expand it, there will be a **node for every function** in the script. Clicking them show the code of a function.
  - There is a new **info node** as well, it shows file information as well as some statistical code information.

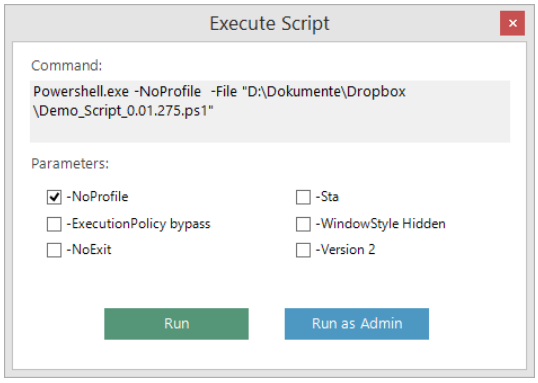


  - You can set the **root** node (topmost folder) for CodeManager for in two ways:
    - Right click on <u>any</u> **location-root** or **folder** in **CodeManager** and choose "*Set as Root…*"
    - Right click on any **folder** in **CodeManager** and choose "*Reset **CodeManager** Root*" to reset to the system standard snippet path.
  - **CodeManager** supports **multiple locations**. Simply add via the "+" button in the menu bar or via the context menu of the "snippets"-node
    - Keep in mind, that you want CodeManager always be as **performant** as possible. The more locations you add, the longer search and re-indexing takes (despite its flash-like performance ☺). Adding and removing locations is quick and easy, so keep only the locations you need in CodeManager.
  - If you **right click** on a snippet or script you have these options:
    - Show the selected snippet or script file in windows explorer
    - Open a snippet in Snippet editor, i.e. to make quick changes
    - Open a script in an external editor of your choice
  - If you **double click** on a snippet or script CodeManager can either:
    - Do nothing / Expand the Function and Info nodes
    - Open Scrips and modules in an **external editor** of your choice
      (tested with ISE, Primal Script, Notepad++. It should work with any editor capable of handling PS1 files.)

- ## Code Window



  - The Code window can now be **resized** now and got a maximize and a minimize/hide button.
  - The new "a<-"-buttons turns **word wrap** on or off.
  - You can **search inside** the code window
    - To use a **regular expression** either click on the regex-button ("*Rx*") in CodeManager or add the prefix "`r:`" in front of the search term.
    - Use the Next/Last buttons to view the results **one-by-one** or click on the "*`No x of y results`*" label to **highlight all search results**.
  - The pencil-button activates the **script-edit mode**. The corresponding save button is only visible while the edit mode is active.
    - This edit mode is not available when you clicked on a **function node** to exclusively view the code of a function. Select the complete script instead.
    - If a **snippet** is selected, a click on the pencil will open Snippet Editor.
  - With the play button, you can **execute the script**, either with admin privileges or as user.



- ## Quick Snippet Creation
  - You can select any code inside the code window and either click Ctlr+N or the respective option in the context menu to quickly create a snippet from that selection.

- ## Quick Search (CodeManager)
  - Whenever you open CodeManager, the **quick search**-box always has the focus, so you can start typing immediately if you like to search for a snippet.
  - Aside from snippets, CodeManager searches per default in script and modules **names**.
  - To search in **script code** as well, activate the "paper scroll"-button

The option can impact search performance, depending on the type of your added locations and number of scripts. (But, to brag a little ☺: if your scripts are on a fast local

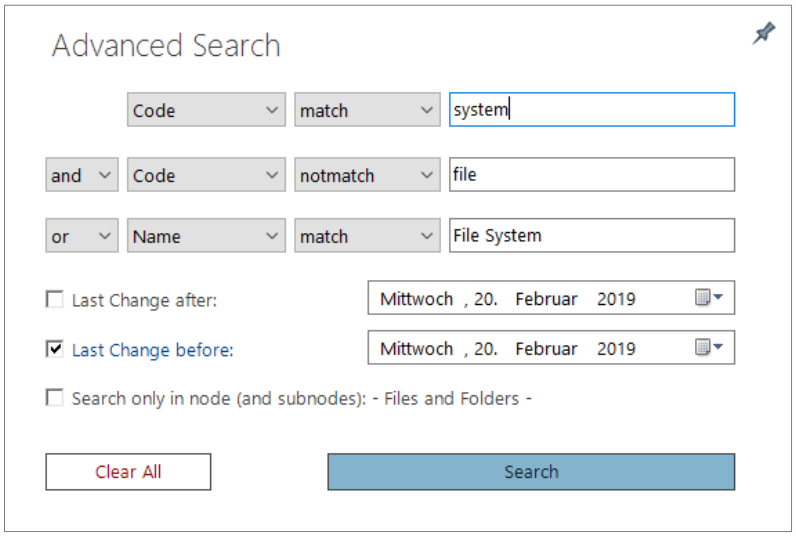
drive, searching through even thousands of large scripts takes only a few seconds. But on network or slow local drives, it obviously takes longer) Some Antivirus-programs increase search time massively as well.

- To search using **Regular Expressions**, either click on the regex-button ("*Rx*") or add the prefix "`r:`" in front of the search term

- **Advanced Search (CodeManager)**

  - To get to the advanced search dialog, hit the magnifying glass -button.

  - In the advanced search dialog, you can build your search query more easily. You have multiple options, but they are rather self-explanatory:



- **Inject / Paste Code**

  There are **three ways** to get code from CodeManager into your script:

  - **Auto Copy to Clipboard - Auto Injection**

    - When you open **CodeManager** and **click** on a snippet or script, CodeManager makes this snippet ready for auto-injection, the Auto-CodeManager gets "**loaded**".
      The little round **indicator** at the bottom of the window turns **red** to show you that.

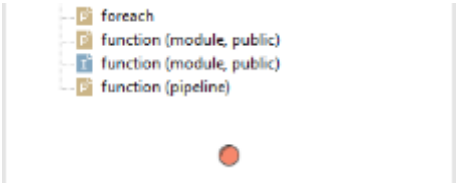
    - You can also **select any code** inside the code window (if you want to use just a part of your snippet or script) and the "Injector" gets **immediately loaded with that selection**.

    - The **next click** you do **anywhere** **outside** CodeManager, is the place where **CodeManager** tries to inject the snippet into.
      So this place should be your editor and the position where you want the new code to end up.

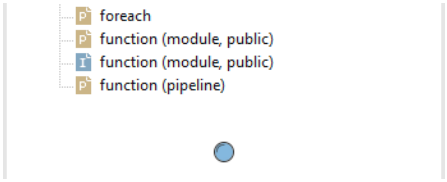
      Please note: *Due to windows security restrictions, **CodeManager** can only auto-inject into applications that run in the **same user context** as CodeManager.*
      Usually that won't be a problem, but if you start your editor with admin rights for example, you need to start CodeManager with admin rights as well, otherwise auto-injection won't work.

    - The snippet code gets also **copied to the clipboard**, so if you have a snippet (e.g. a function) that you want to inject a couple of times, you can by just using **CTRL-V**.

- If you clicked on a snippet but don't want to auto-inject, you can manually "**unload**" the auto-injector any time, by either **clicking** on the red **indicator** or on **any folder** in CodeManagers explorer.
- NOTE: Because this behavior may take some time getting used to (it is worth it), **auto-injection is deactivated** (options menu) **by default.**



A) **CodeManager** ready for Auto-Injection



B) Snippet or code selection has been copied to Clipboard

- **Auto Copy to Clipboard - Manual paste**
  - Is the **standard behavior** (change in CodeManagers options menu)
  - When you **click** on a snippet or script, its code gets **copied to the clipboard** and the indicator at the bottom gets **blue**.
  - You can also **select any code** inside the code and it gets **immediately copied to the clipboard**.
    Yes, you can right-click and select "copy", but it's not necessary. ☺
  - You can then paste it with the usual **CTRL-V** anywhere you like.
- **Manual Copy to Clipboard - Manual paste**
  - If you don't want that all shown or selected code gets copied to the clipboard, you can quickly deactivate this as well, with the clipboard button in CodeManagers menu bar:

4. Snippet Backup

With the new backup function, you can create a **single file backup** of all your snippets in the **systems snippet path (see below)**. This can be done **manually or automatically** every time Code Manager exits.
Apart from the obvious benefits of a backup, this can be used to easily and quickly **get your snippets onto a new machine** or even **sync changes** between machines.

- **Open** the backup dialog with the diskette button in the main window.

4.1 Backup

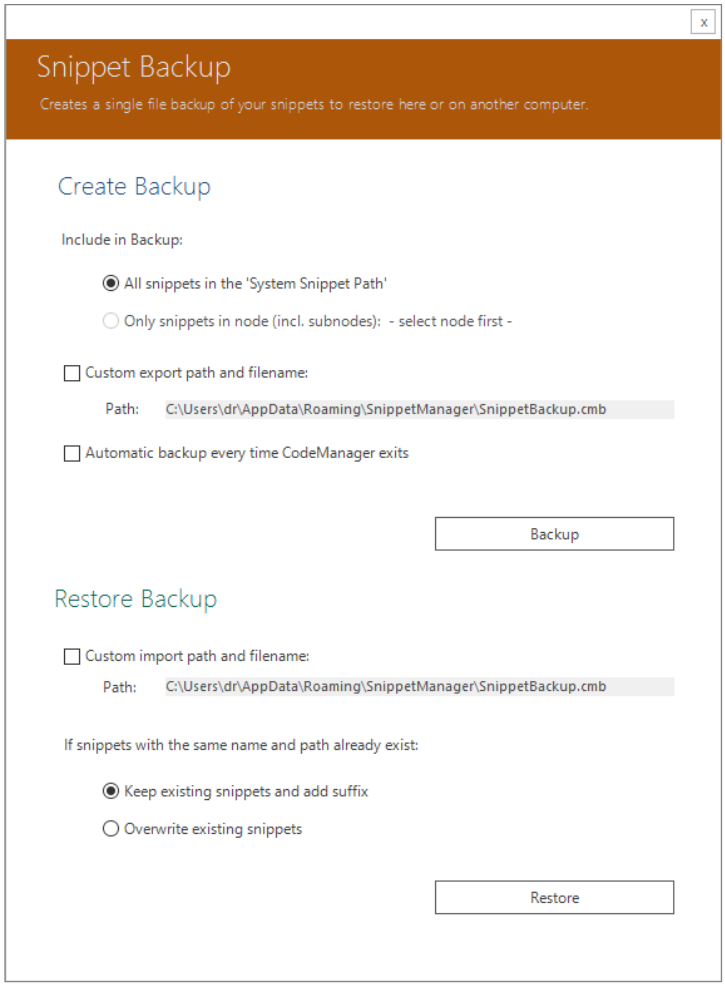
- Creates **manually** or **automatically** a **.cmb*-File (which is basically a ZIP), at the location of your choice, with your snippets.
- CodeManager also calculates a SHA256-hash of the backup file, which be be written in an *.hash file in the backup target directory. It will be later used for auto-sync.

4.2 Restore

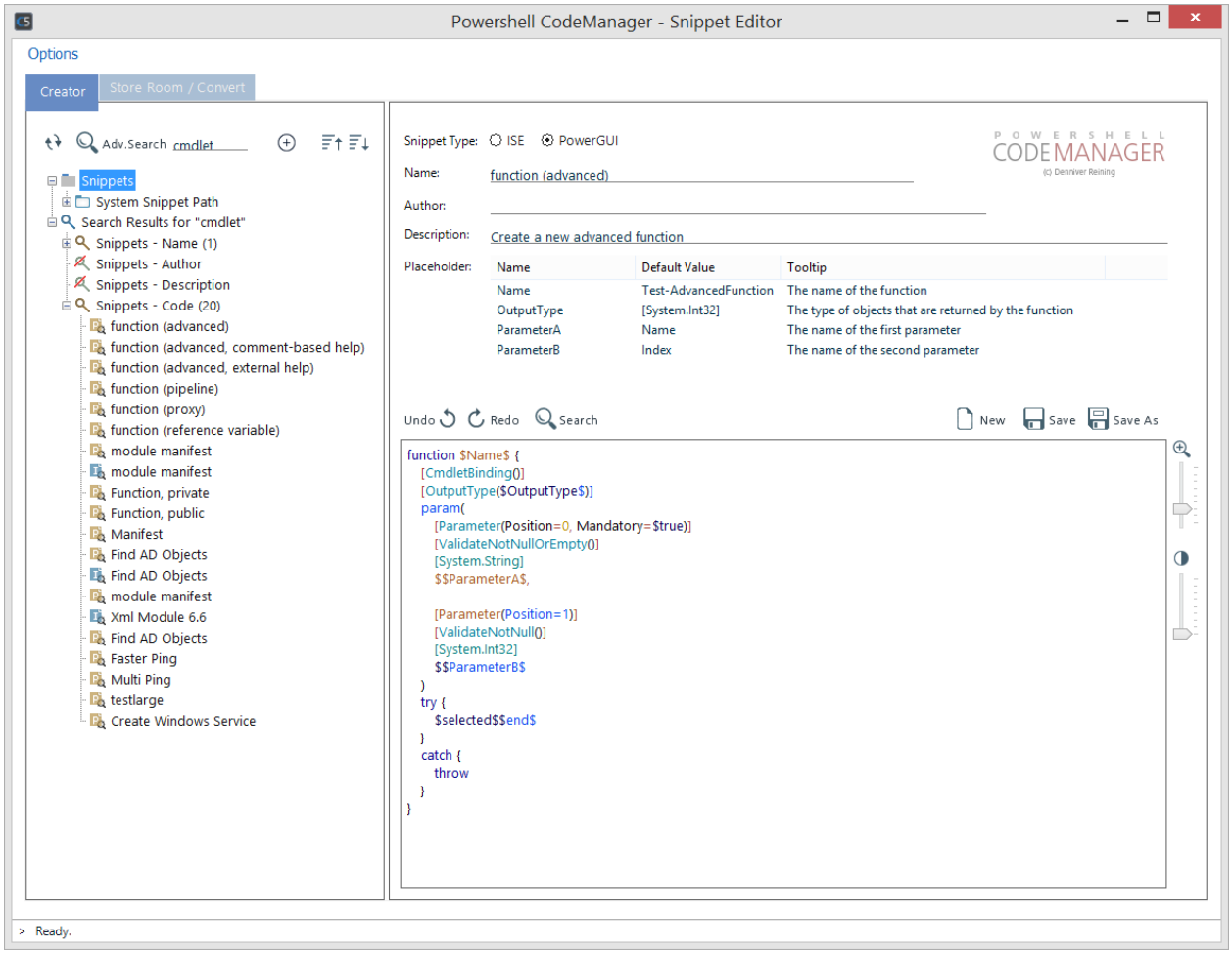
- You can restore a snippets backup from any machine to your system snippets path.
- If you choose to **keep** existing snippets with the same name, they will be renamed and get a "*_old*" **suffix**.

4.3 Targeted Backup / Share Snippets

- Right click on any folder to create a backup file to a location of your choosing
- Note that although any files in the selected folder (and subfolders) will be included in the backup, only Scripts, Snippets and Modules can be restored via CodeManager.



5. The Snippet Editor



4.1 The Snippet Explorer

- The snippet explorer is on the left side of the window. Here you have standard explorer functionality (Copy, Paste, Delete, Rename etc.) except for one limitation: no multi-selection of items.

  In order to compensate for that, there is the "***Show in Windows Explorer***"-context menu item, where you can go directly to the item or folder of your choosing and make any kind of mass actions via Windows Explorer. If you get back to CodeManager just click "***Refresh***" to see your changes. ISE snippets will be displayed with a blue icon with an "I" and PowerGUI snippets with a beige icon with a "P".

- To **open a snippet**, just double click on it.

- You can **add** other snippet folders ("**Locations**") to the snippet explorer by clicking "Add".

  - You should **add folders with snippets** only, instead of adding e.g. whole drives, because this would slow down file operations considerably.

  - If you add a location you have no write access to, the location-icon will have a bars in front of the folder icon to remind you of that fact. :)

  - You can also add UNC-paths manually, via the "Add" window.

  - **Note**: CodeManager use their own set of locations. So a location you add in Snippet Editor does not show up automatically in **CodeManager**. This is for you to be able to keep CodeManager always only as stuffed as necessary and therefore as performant as possible.

- **Remember**: ISE and PowerGUI load their snippets (per default) only from the standard snippet path. (see chapter 6.) You manage snippets with CodeManager on any path, but in order to use them you need to copy them to this path.
  Both ISE and PowerGUI have a limited option to use snippets from other paths, please consult the documentation for that.
  But, if you use CodeManager for inserting snippets, **ignore** what you've just read, because you can use any path from any location. ☺

- You can **remove a location** by right-clicking on the location root and choosing "Remove..."

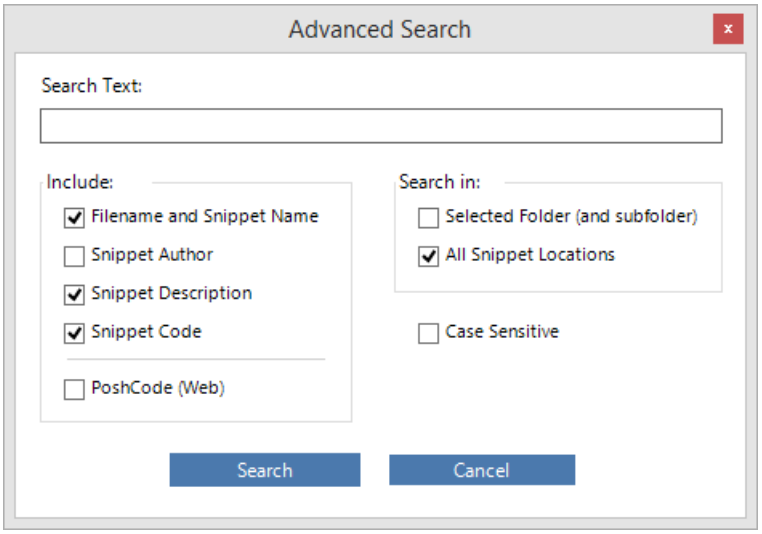
- You can **set the root** (first or parent folder) **for CodeManager** window, by right-clicking any folder or location in CodeManager and choosing the menu item "Set as Root...."

4.1.1 Search for Snippets (Snippet Editor)

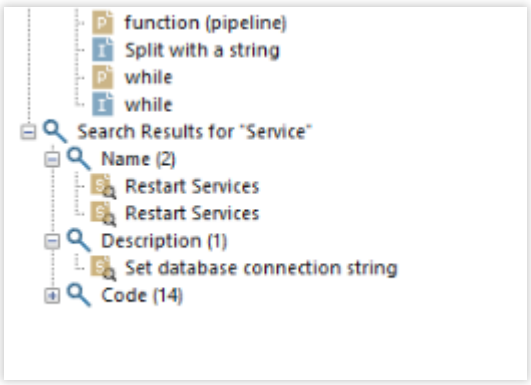
NOTE: Unfortunately, PoshCode.org does not exist anymore, so the search has been removed with 6.0.

- Via the **Quicksearch**-box, you can search in every part of every snippet the snippet explorer. Just enter your search text and hit "ENTER". QuickSearch always searches in all locations, all fields and case insensitive.

- Via the **"Adv. Search" - button**, you can set options for your search



- Search results will be displayed as a new tree in the explorer:
- You can now further filter search results by name and creation date, via the context menu for each search result node (or the root node).

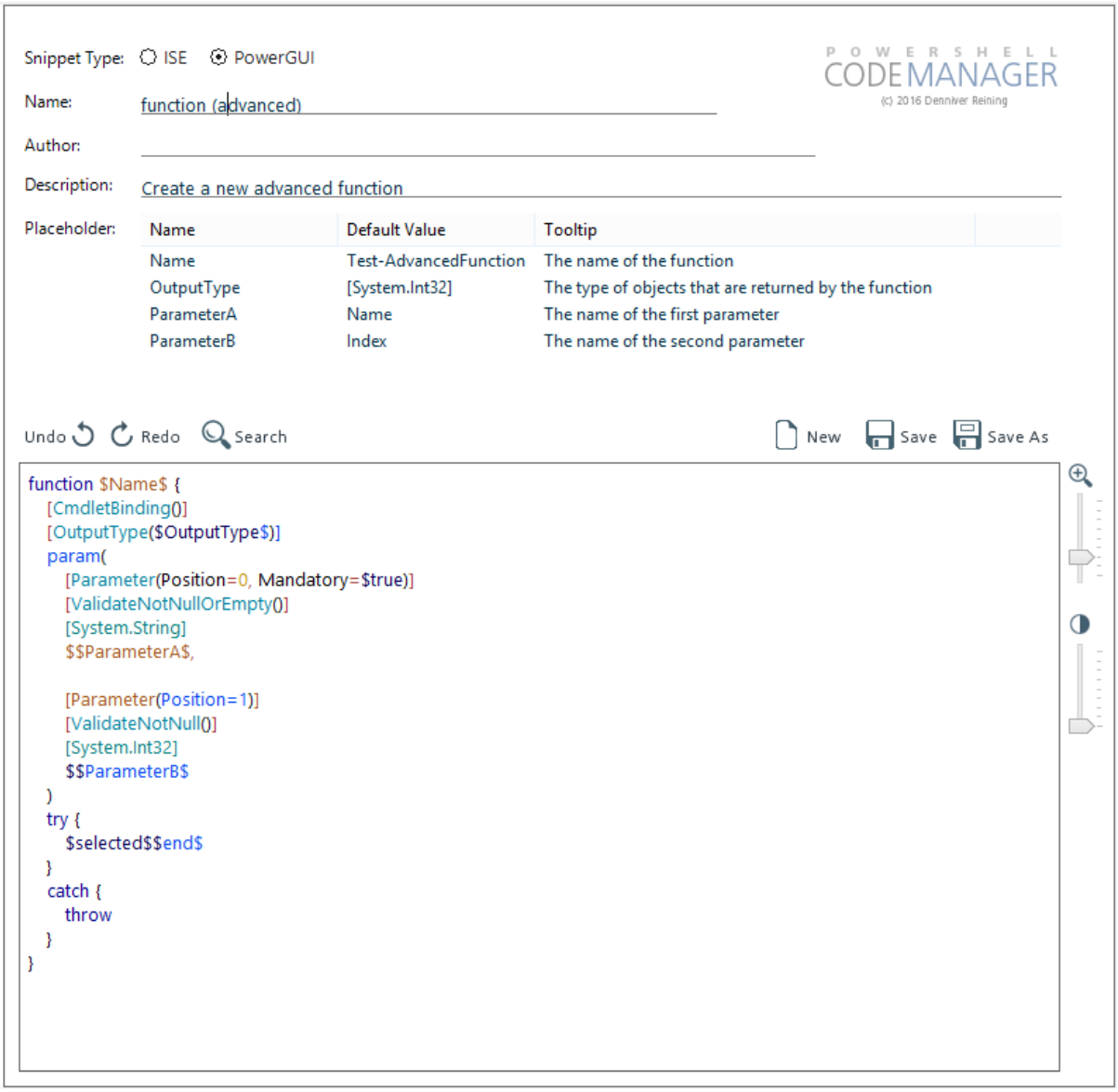


4.2 File Names, Snippet Names and Folders

- **File and Snippet Names**
    - With v4 I removed the annoying difference of file and snippet names. As long as you create and edit your snippets with CodeManager, file and snippet name always stay the same.

- **Folders**: The folders that are shown (and can be created and edited) in the snippet explorer, **also act as categories** for the snippet menu in PowerGUI. A good way to group your snippets and keep the snippet menu tidy.
  ISE does not support folders/categories. ISE does read all snippets from all folders in the snippet path though, but presents them flat.

4.3 The Snippet Window



- The **Snippet Type** shows if the snippet is in ISE or PowerGUI format. You can change the type any time you want and then save the new snippet format. If you want to get rid of the old one, you can to delete it in the snippet explorer.
  To convert more than one snippet at a time, use the Snippet Converter on the "Store Room/Converter" – tab.

- The ***Snippet Name*** is what you see in the snippet menu in PowerGUI, its mandatory for every snippet.

- ***Author and Description*** are optional, but a description is a good idea, it shows up as tooltip in the snippet explorer, and as well in the snippet menu in PowerGUI and ISE.
  TIP: You can add a **standard** value for the **Author** field under "*Options/Auto Author" and e*very time you hit *"New",* the Author-field will be filled in automatically.

- The **Code box** is where the snippets code sits. (surprise ☺)

- The **search button** lets you search for anything in the code box and every found instance gets highlighted.

- The ***Undo***, ***Redo***, ***Save*** and ***Save As-*** Buttons won't need explaining. With the ***New*** - button you can clear the snippet window in order to create a new snippet.

4.6 Markers

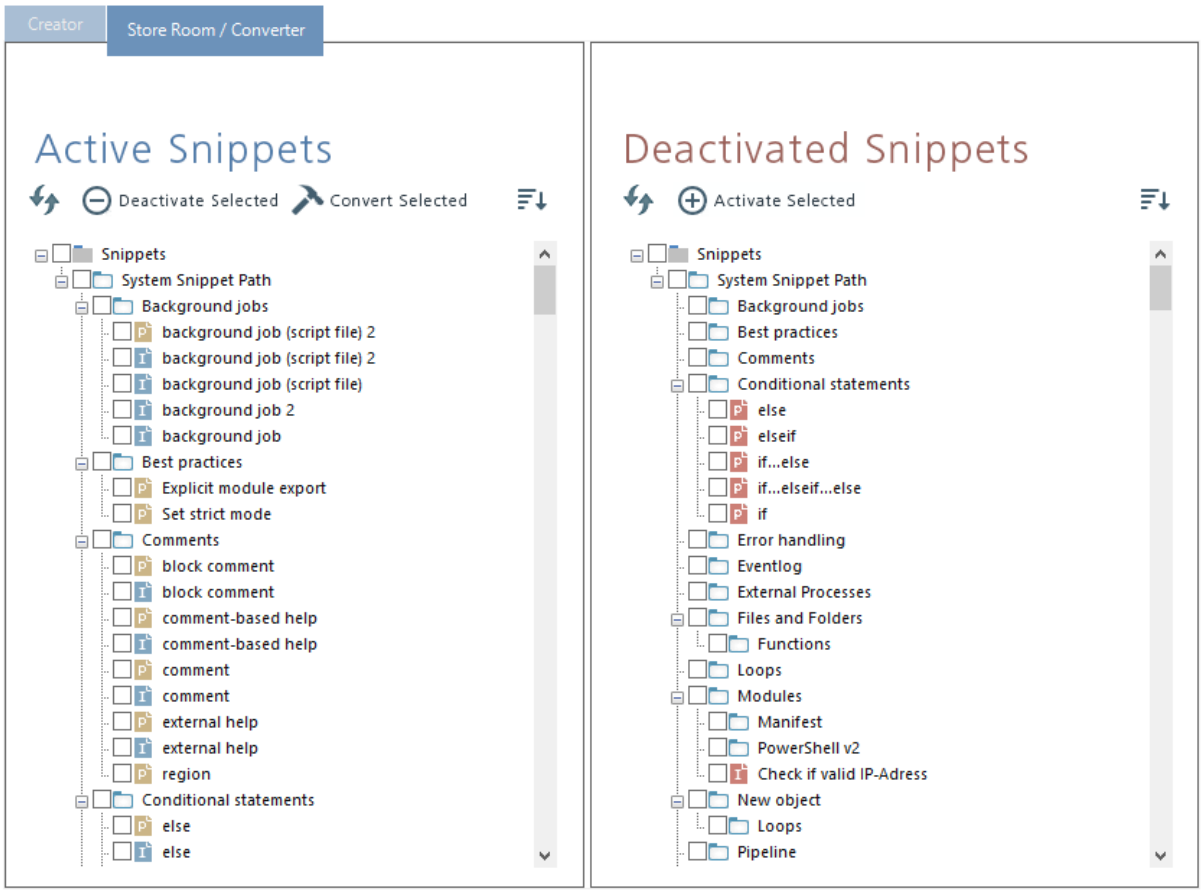
There are two special placeholders: **END** and **$SELECTED$** which you can **insert from the context menu** of the code box.

ISE only supports the end-marker, **PowerGUI** both end and $selected$-marker.

(And although you could just type them in, I suggest you do it via the context menu, because in this case CodeManager automatically **checks** if you already have the chosen marker somewhere in the code and if yes, **highlights** it for you.)

- The **END** marker marks the place where your cursor sits after you inserted the snippet to your script.
- When you select something (a string, a command, whatever) in your script and then insert the snippet, the $**SELECTED$** marker will be replaced by the selected text. In other words, the snippet will be "wrapped around" your selection.

6. The Store Room / Converter (activate, deactivate and convert snippets)



5.1 Activate / Deactivate

In the Store Room you can **deactivate (and reactivate) snippets** you don't need at the moment, so that they don't take up space in (the rather small) *"Insert Snippet"*-menu in **ISE** or PowerGUI.

- If there are e.g. shipped snippets that you probably never use but you don't want to delete them, or you have sets of snippets for different projects, just **check** them (separately or the whole folder) **and click** "Deactivate Selected" on the left side.

  ISE:

  The snippets which are shipped with ISE can't be deactivated here, because they are not actual snippet files. But they can be deactivated via the "Tools\Options"-menu entry in ISE

- To **reactivate** snippets, just do the same on the right side and click "Activate Selected".

- If you deactivate **all snippets in a folder**, the folder itself will **still show up** everywhere in CodeManager, but in PowerGUI it won't. (ISE doesn't display folders at all)
  This is because **deactivated snippets stay where they are**, they will just have a different file extension so that PowerGUI ignores them.
  So don't be surprised, if you try to delete a seemingly empty folder and CodeManager warns you that there are deactivated snippets in it.

5.2 Conversion

On the left side, there is also the **"Convert Selected"-**Button. With this you can convert any snippet (PowerGUI or ISE) into the other format. It's simple:

- Just select all snippets (ISE, PowerGUI or both; single files or complete folders) you like to convert and hit **"Convert Selected"**
- You can then **choose if you like to keep** the originals (recommended!) **or delete** them automatically. Keeping is recommended, because first, they don't bother you. Each editor shows only his type of snippets. Second, if you convert snippets from PowerGUI to ISE and then delete the PowerGUI snippets, you lose the placeholders stored in them. If you really don't want them to show up in the snippet explorer, I recommend rather deactivating them in the Store Room.

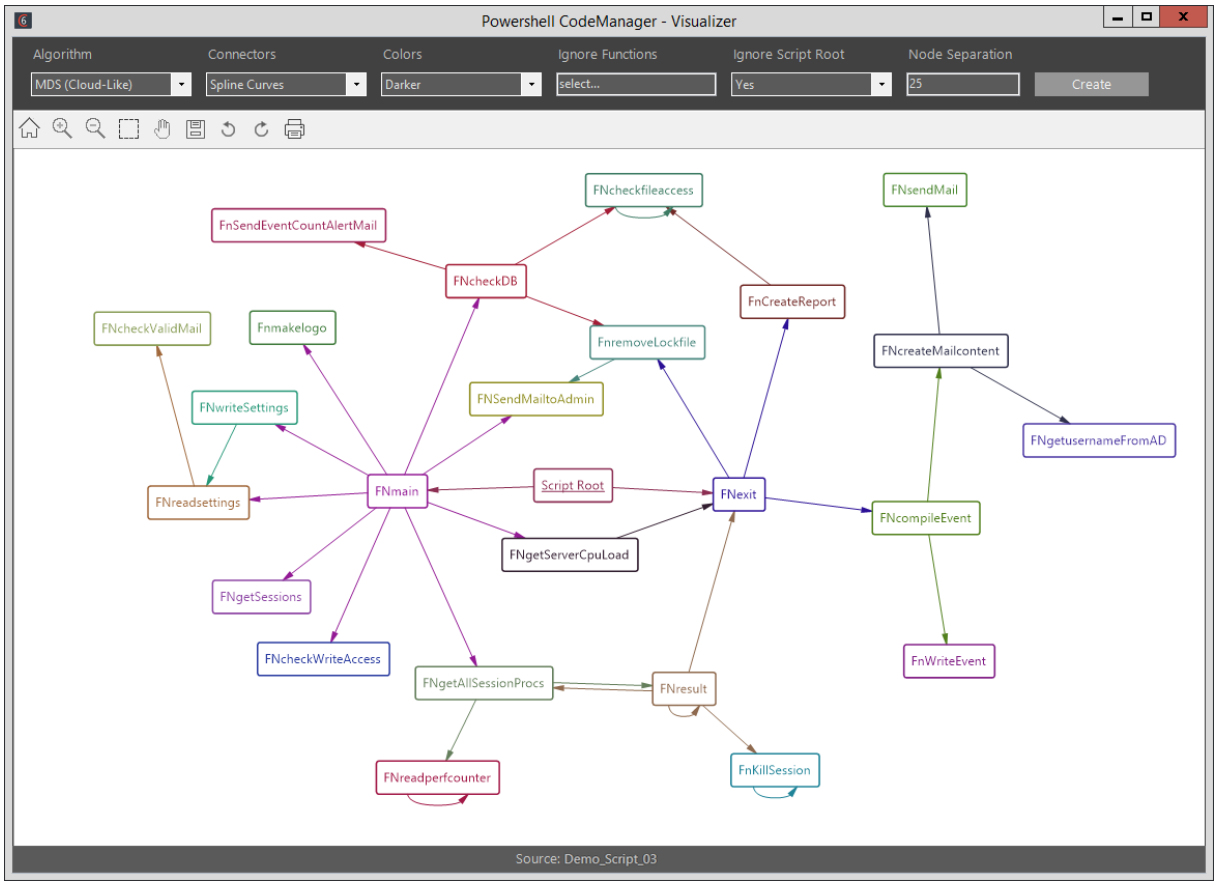
Btw.: CodeManager **does not care about the snippet type**, no matter what the target editor is.

7. Visualizer

The Visualizer creates **interactive function dependency graphs**. That means it displays all the functions of a script and draws connection lines between them, to depict how they interact with each other and from where they are being called.

Interactive
Each element of the graph can be clicked to show either the code of the selected function or the exact position from where a function is being called.

Documentation
Graphs can be printed and saved either as PNG image or in vector format to allow further editing in any vector capable graphics tool.



Creation

- Graphs can be created for **complete scripts** (all functions and function calls) and **single functions**, to display only the calls made *by* this function and calls made *to* this function.
- Graphs can be created by:
    - Right-click on every **script, module or function-node** in CodeManagers explorer and selecting “Create Function Dependency Graph….” In the context menu
    - Right-click on a **node** of a **currently displayed graph** to either create a graph for the function you clicked on or right click **anywhere** in a graph to go back and display to complete graph for the script again.

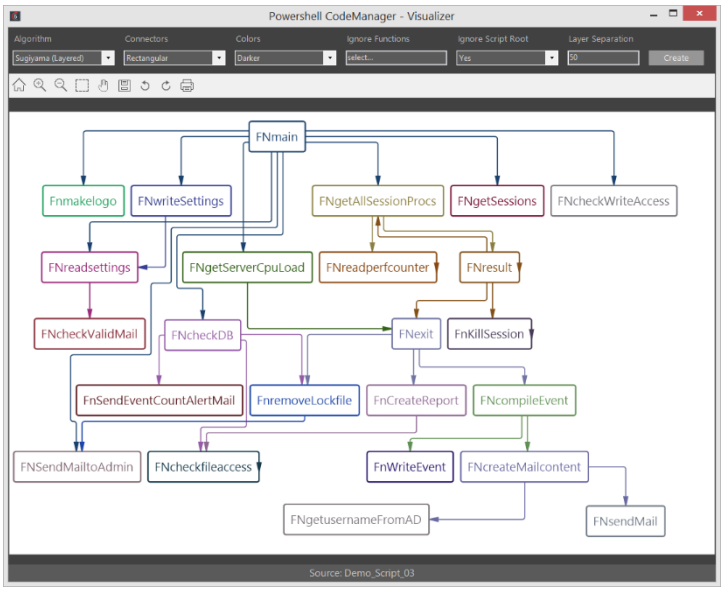
- Opening Visualizer with the button in CodeManagers menu bar, then selecting any snippet, script, module or function in CodeManager and then using the "**Create" button** in Visualizer

Visualizer Button

- No matter which way you create a graph, **Visualizer always uses the settings** currently set in the settings-section at the top of the Visualizer window.

- If you create a graph with the ***"Create"*-button**, Visualizer always uses **the currently selected node in CodeManagers explorer** as source for the graph. No matter what is currently displayed in Visualizer.

Settings



- **Algorithm**
  The algorithm determines how the nodes will be placed on the graph. You can currently choose between Sugiyama and MDS. **Sugiyama** draws nodes in **layers**, while a **MDS** graph looks more like a **cloud** of nodes.

- **Connectors**
  Usually Spline connectors fit better to MDS and rectangle connectors better to Sugiyama, but that depends on the complexity of the graph and the number of nodes.

- **Colors**
  The color schemes should be self-explanatory. **Please note** that in the schemes "*Full Range*" and "*Darker*", the colors get assigned by **random**. So if you don't like a particular color assignment, just render the graph again with the "create"-button to get different colors.

- **Ignore Function**
  If you have a complex graph and there is one function that has a particular high number of connectors, it can make the graph messy. Especially if this particular function is a rather unimportant one, like a logging function. So, you can simply ignore that function or others by selecting it them the list here.
  You can also right-click on every node in a graph and add that node to the ignored-list.

- **Ignore Script Root**
  If "yes" is selected here, any function call which is placed not in another function, but in the root level of the script, will be ignored.

- **Node/Layer Separation**
  Depending on the selected algorithm, you can either set the minimum distance between nodes for the MDS algorithm or the horizontal distance between the layers of the Sugiyama algorithm.

Exploring

Function dependency graphs are a great way to explore and understand a complex script, written by someone else.

- Double click on a **function node** to see the code of the function.

- Double click on a **connector** shows the exact position the function call is located in the script.

- NOTE: The code window **disappears automatically** if you move the mouse out of it or if you click anywhere in the graph. If you like it to stay open, use the PIN-button in the code window.

- Right click on a function node to "**zoom** in" and create an isolated graph of this function and its dependencies.

Editing

- By single clicking on a node, you can move it around to e.g. correct an odd placement or if you simply like the graph to look different. You can also select multiple nodes (by drawing a box like in windows explorer), and move them around in groups. The related connectors will be redrawn automatically.

Saving

- Graphs can be saved either as:
    - **Image**. You can choose the **image size** in pixels. Pixel based images should not be scaled up later, because they lose quality. As a rough guide: For including a graph in a **word** or **pdf** file, choose a size about roughly your screen-size. For **printing**, choose three times your screen size. But be careful: very high pixel counts can create huge files.
    - **Vector** format files (*.svg). SVG files are very **small**, can be opened by any vector-capable graphics program (LINK), **scaled** without quality loss and are fully **editable**.

Printing

- The print function is still beta, so if you run into any trouble, just save the graph as an Image and print the image with your picture viewer.

8. About the snippet paths

CodeManager

If you choose to use CodeManager to insert snippets instead of ISE or PowerGUI snippet insert menu or do not want to use snippets at all, you can **ignore this chapter**. You can insert code from any source into your editor.

ISE (V3 and higher)

Your snippets are stored in:

- *<mydocuments>\WindowsPowerShell\snippets*

POWERGUI

PowerGUI stores its (shipped) **snippets** in a subfolder of its **installation folder**. Usually, that is:

- *C:\Program Files\PowerGUI\snippets*

As a **security measure** (since Windows Vista), no program (or user for that matter) without admin privileges is allowed to write to the programs dir. If you try it by hand, you get an elevation prompt. If a program tries it, on some systems you get an error, on some systems the *folder redirection* kicks in and redirects every change you made to a folder called virtual store. **Since** version **2.3**, PowerGUI uses a **second snippet path** per default:

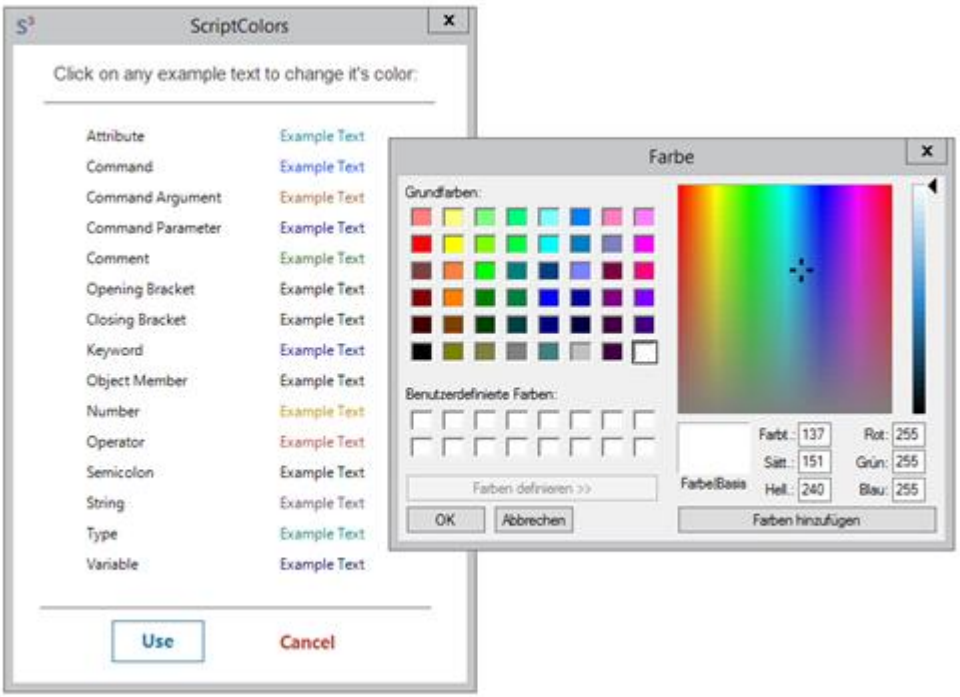
- *<mydocuments>\WindowsPowerShell\snippets*

This path, being in the **user profile**, is fully accessible to the user. So when you first run CodeManager, it asks you if you want to move the snippets from the installation folder to the user profile. You then have the following **options**:

- You can **move them**. That means:
  - You will be able to **edit and deactivate/reactivate all of the snippets**.
  - Only the user profile snippet path will be added automatically to the snippet explorer in CodeManager.
  - If CodeManager detects snippets in the PowerGUI install directory (e.g. after an PowerGUI update) you will be asked again.
- **Or** you can **leave them** in the installation folder. That means:
  - Both paths (install dir. and user profile) will be added automatically to the snippet explorer in CodeManager and **you can only edit or deactivate/reactivate the snippets in the user profile**.
  - **But** and this is very important(!), **both paths will be used by PowerGUI** when you open the *"Insert Snippet"*-menu. If there are snippets with equal names in both folders, the one in the user profile will **take precedence** over the other one.
  - You won't be asked again about this. But if you should later decide to move the snippets after all, I recommend not doing it by hand, but by clicking the menu item "*HELP\Reset all settings...*", because in this case you want CodeManager to watch the "install dir/snippets"-folder for changes in the future.

9. Custom Syntax Coloring

- Syntax Coloring can be configured via the option menu in CodeManager
- Aside from the two shipped **coloring schemes** (ISE standard color scheme, CodeManager color scheme) you can create your own scheme. Fine tune the existing schemes or create a completely new one
- Performance and Language Problems
    - For very large scripts, syntax coloring is currently deactivated. *(If you are curious: This is due to a very annoying limitation in RichTextBoxes, which lets them freeze for up to a couple of minutes when trying to display a large amount of formatted text.)*
    - If you are on a slow computer, you can also deactivate syntax coloring for mid-size scripts
    - There has been a problem reported Syntax Coloring currently has with Cyrillic characters, so as a workaround, Syntax Coloring can be deactivated completely inside the options menu



10. Addendum

Keyboard Shortcuts:

- ### Snippet Editor

Code Window:	
F3	Search in Code
F5	Reparse Code
F6	Auto Create Placeholder
F7	Half-Auto Create Placeholder
CTLR + S	Save Snippet
CTLR + SHIFT + S	Save Snippet As...
CTLR + N	New Snippet
In Snippet Explorer:	
F3	Advanced Search
F5	Re-Index Snippet Folder
CTLR + F	Advanced Search
CTLR + C	Copy
CTLR + X	Cut
CTLR + V	Paste
ENTF	Delete
Return	Load Snippet

- ### Code Manager

Code Window:	
F3	Search in Code
F5	Reparse Code
CTLR + F	Search in Code
CTLR + N	New Snippet From Selection
Script/Snippet Explorer:	
F3	Quick Search
F5	Re-Index Currently Selected Node
CTLR + F	Quick Search
CTLR + N	New Snippet
CTLR + E	Execute Selected Script
CTLR + R	Rename
CTLR + C	Copy
CTLR + X	Cut
CTLR + V	Paste
CTLR + G	Create Function/Script Graph
ENTF	Delete
Return	Show Code of Selected Node