# Lab 6 – PowerShell 2

**Lab 5 must be completed before starting this lab.**

This lab is to be completed on **Server1.**

Approximate completion time: 75 minutes

Upon completion of this lab you will be able to:

* Install Features
* Import Modules
* Create Functions
* Edit your profile to include Import-Module and Functions
* Use PowerShell ISE to create a script
* Run a script from the File Explorer, PowerShell Window and PowerShell ISE

## 0.0 Installing Hyper-V and Hyper-V PowerShell Features

1.Open PowerShell using your custom shortcut from the previous lab.

2.Type the following commands to install the Hyper-V and Hyper-V PowerShell Features to your server:

***Add-WindowsFeature Hyper-V-Tools***

***Add-WindowsFeature Hyper-V-PowerShell***

## 1.0 Import-Module

A module is a package that contains PowerShell commands, such as cmdlets, providers, functions, workflows, variables, and aliases.

People who write commands can use modules to organize their commands and share them with others.

After a module is imported, you can use the module members in your session.

1.Some modules come pre-imported (installed). Some modules are on the server but need to be installed before you are able to use the cmdlets and function in them. To find the modules that have already been imported into your session, at the PowerShell prompt, type:

***Get-Module***

2. To find modules that are installed in a default module location, but not yet imported into your session, type:

***Get-Module -ListAvailable***

3. To import a module, use the Import-Module cmdlet. Use the following command to import the Hyper-V module:

***Import-Module Hyper-V***

4. To see a list of commands in a module, type:

***Get-Command -Module Hyper-V***

5. To get help for any commands in the module, use the Get-Help command:

***Get-Help New-VM***

6. While there is really no point in removing a module, since the imported module will be lost as soon as you close this PowerShell session, the command to remove a module is:

***Remove-Module Hyper-V***

**\*\*Take a screen shot of the PowerShell console showing the output of the   
G*et-Command -Module Hyper-v* command. Name this file *PS-module.jpg***

## 2.0 Creating a Function

A function is a list of PowerShell statements that has a name that you assign. When you run a function, you type the function name. The statements in the list run as if you had typed them at the command prompt. It is like a mini-script or a complex cmdlet with parameters and options.

1.Type the following command to create a function that starts PowerShell with the Run as Administrator option:

***Function Start-PSAdmin {Start-Process PowerShell -Verb RunAs}***

2. Run the new function by typing:

***Start-PSAdmin***

A new Administrative PowerShell window should open.

3. At the PS prompt, type the following multi-line function called Get-SmallFiles. This function has a user-defined variable, named $size. The function displays all the files that are smaller than the value of the $size parameter, and it excludes directories:

***function Get-SmallFiles {***  
***Param($Size)***  
***Get-ChildItem | Where-Object {***  
***$\_.Length -lt $Size -and !$\_.PSIsContainer***  
***}***  
***}***

4. To run the new function, type:

***Get-SmallFiles -Size 50***

**\*\*Take a screen shot of the PowerShell console showing the output of the *Get-Smallfiles*function. Name this file *PS-function.jpg***

## 3.0 Editing your profile

The Imported module and functions from the previous 2 sections are only valid for the session in which the module was imported, or the function was created. To make these work for all your PowerShell sessions, you can add them to the profile you created in Lab 5.

1. Edit the profile by typing:

***Notepad $profile***

2. When notepad is open, enter the Import-Module command from Section 1.0 and the 2 functions you created in Section 2.0. Save and exit Notepad.

3. Close your PowerShell Session and open it again. To prove your profile works, type the following:

***Get-Module***

You should be able to see Hyper-V in the list of modules.

Your alias from the previous lab should be available and the functions you created.

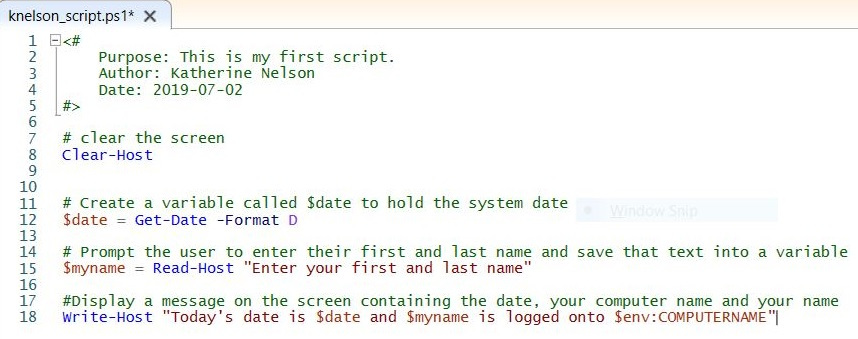
**\*\*Take a screen shot of the PowerShell console showing the output of the *Get-Module* command. Name this file *PS-profile2.jpg***

## 4.0 Creating your first script

1.Open PowerShell ISE.

2. Open a new script pane by clicking the File menu and selecting New.

3. Type in the following script, substituting your name and the current date:



**While typing in the commands, try the INTELLISENSE feature, by typing parts of commands and use the pop-up suggestions.**

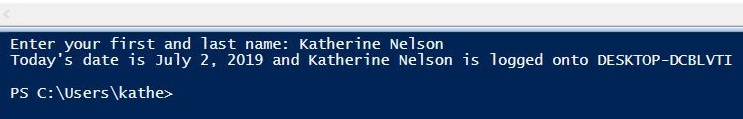
4. Once you have the script typed in, save the file as SenecaID\_script.ps1.

5. Run just the Get-Date command by highlighting ***Get-Date -Format D*** and click on the run selected button at the top of the screen. You can do this with any part of the script.

Graphical user interface, application, Word

Description automatically generated

6. Run your entire script. You should see output similar to the following:



## 5.0 Changing the flow of your script

1. Edit your script to include a variable to hold a random whole number and to prompt the user to enter this number.

2. Using the examples from the lesson as a guide, write an IF/ELSE statement that tests the value of your variable from #1 and displays an additional message on the screen depending on the number entered.

3. Run your script.

**\*\*Take a screen shot of the PowerShell ISE console showing your script and the output of the script. Name this file *PS-myscript.jpg***

**To prove you have completed this lab:**

* **Create a Microsoft Word document (or use Google docs), with a name of *YourSenecaID*-Lab6.docx.**
* **Paste each screen shot from the lab into the MSWord document, and label them clearly. (PS-module.jpg, PS-function.jpg, PS-profile2.jpg, PS-myscript.jpg)**
* **Save the document as a PDF file (or Print as PDF) using the same name as the document file and upload it to Learn@Seneca. You will find the link in the Weekly Module on Learn@Seneca.**