



# Introduction to PowerShell

Module 1

# Learnings Covered in this Unit



System Requirements for PowerShell



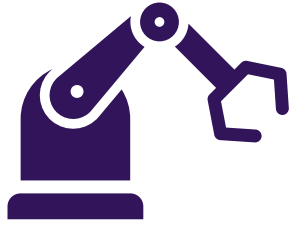
PowerShell Shell



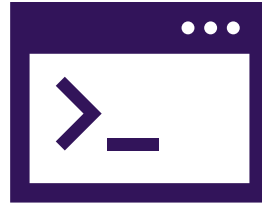
Integrated Scripting Environment

# What Is PowerShell?

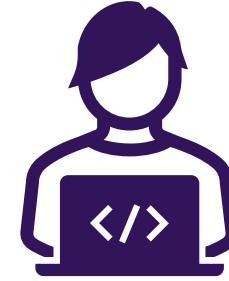
# PowerShell Overview



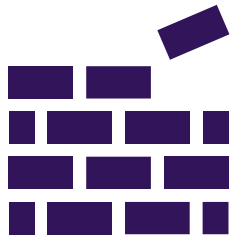
Task **automation**  
framework



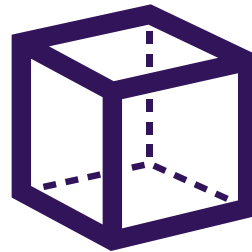
Command-line  
**shell**



**Scripting**  
Language



Built on **.NET**  
Common  
Language Runtime  
(CLR)



Accepts and  
returns .NET  
**objects**



Cross **platform**  
(v6+)

# Modern PowerShell Versions

## Windows PowerShell 5.0

- Released: 2015
- Package Management
- **PowerShell Get**
- CMS module
- Remote debugging
- DSC additions



## PowerShell 6.0 Core

- Released: 2018
- **Windows**
- **Linux**
- **Mac OS**
- Docker support
- SSH remoting
- Pipeline commands in background



## PowerShell 7.0

- Released: 2020
- Improved **compatibility** with Windows PowerShell
- Pipeline chain operators
- Parallel execution added to ForEach-Object

# System Requirements

5

**Default:** Windows 10

**Default:** Windows Server 2016+

.NET 4.5

WMF 5.1 Update

**Minimum:** Windows 7 SP1

**Minimum:** Server 2008 R2 SP1

7

Windows 10

Windows Server 2019

Many flavors of Linux

MacOS

**Minimum:** Windows 8.1

**Minimum:** Server 2012

# PowerShell Hosts

# Hosts

## Command Line Interface (CLI or **Shell**)

- Interactive mode used for simple commands not for scripting

## **Development** Environments

- Graphical script development tool with auto-complete
- Interactive execution of code
- Debugging
- Extensible and customizable

## Integrated Scripting Environment (**ISE**)

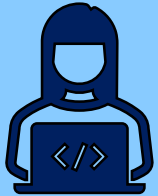
- Pre-installed with windows
- No new development on roadmap

## Visual Studio Code (**VSCode**)

- Cross platform
- Multi-language editor
- Frequent feature updates



# Command-Line Interface with PSReadline



**Interactive** mode used for simple commands not for scripting



Useful **hotkeys**: Up/down arrows, Tab, Ctrl+Space, Undo/Redo



Syntax **highlighting**



Saved **history** of commands between sessions

Windows PowerShell

```
PS C:\> Get-ChildItem -Path
```

Path	Depth	File
LiteralPath	Force	Hidden
Filter	Name	ReadOnly
Include	UseTransaction	System
Exclude	Attributes	Verbose
Recurse	Directory	Debug

```
[string[]] Path
```

Windows PowerShell

```
PS C:\> Get-Process
```

Handles	NPM(K)	PM(K)	WS(K)	VM(M)	CPU(s)	Id	ProcessName
83	7	1112	4224	45		2296	armsvc
184	12	2248	7664	92		1228	atieclxx
109	6	776	3460	23		1012	atiesrxx
461	19	5056	16564	110	0.80	6020	BasisSync
182	13	2236	8200	89	0.11	4808	BDAppHost
142	11	1948	7076	76	0.13	4452	BDEXTHost
315	22	11332	20060	122	0.67	6784	BDRuntimeHost
402	30	7780	26080	207	0.63	6320	BingDesktop
1101	89	91144	7712	859	7.02	6504	CCC
890	43	19424	49124	158		5668	CcmExec
51	6	904	4068	53	0.00	3420	conhost
107	10	3232	10084	101	1.06	4216	conhost
376	24	45200	55396	365	0.58	7356	CSISYNCLIENT
416	15	1824	4200	50		616	csrss
507	28	2300	6208	134		728	csrss
129	5	984	3988	18		2380	dasHost

# Demonstration

PS Readline

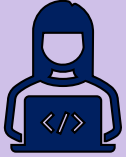


Questions?



# Integrated Scripting Environment (ISE) - Basics

# Integrated Scripting Environment (ISE)



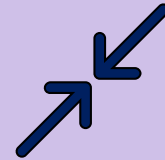
Used for **script** development



**Snippets**



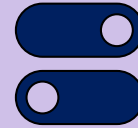
**Customizable** Syntax highlighting



**Collapsible code**



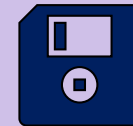
**Error highlighting**



**Brace Matching**



**Debugging**



**Auto-save** and **Crash Recovery**



**IntelliSense**



**Extensible**

# Anatomy of the ISE

The screenshot shows the Windows PowerShell ISE interface with the following components identified by callouts:

- PowerShell tabs**: Located at the top, showing tabs for 'PowerShell 1' and 'PowerShell 2'.
- Scripts open within a tab**: A callout points to the script editor area where a script named 'Mandatory-Param-DEmo.ps1' is open.
- Script pane**: A callout points to the right-hand pane displaying a list of commands, including '1Level-Manual-Serialization.ps1'.
- Console pane**: A callout points to the bottom pane showing the output of the 'Get-Service' command.
- Show-Command add-on**: A callout points to the 'Commands' pane on the right, which lists various PowerShell commands.

The script editor shows the following code:

```
1 Function test
2 {
3     Param
4     (
5         [parameter(
6             Mandatory=$true,
7             ValueFromPipeline=$true,
8             HelpMessage="This param should be populated"
9         )]
10        $Incoming
```

The console pane shows the output of the 'Get-Service' command:

```
PS C:\> Get-Service

Status Name                DisplayName
-----
Running AdobeARMservice      Adobe Acrobat Update Service
Stopped AeLookupSvc        Application Experience
Stopped ALG              Application Layer Gateway Service
Stopped AllUserInstallA... Windows All-User Install Agent
Running AppIDSvc          Application Identity
Running Appinfo           Application Information
```



# Demonstration: ISE Anatomy



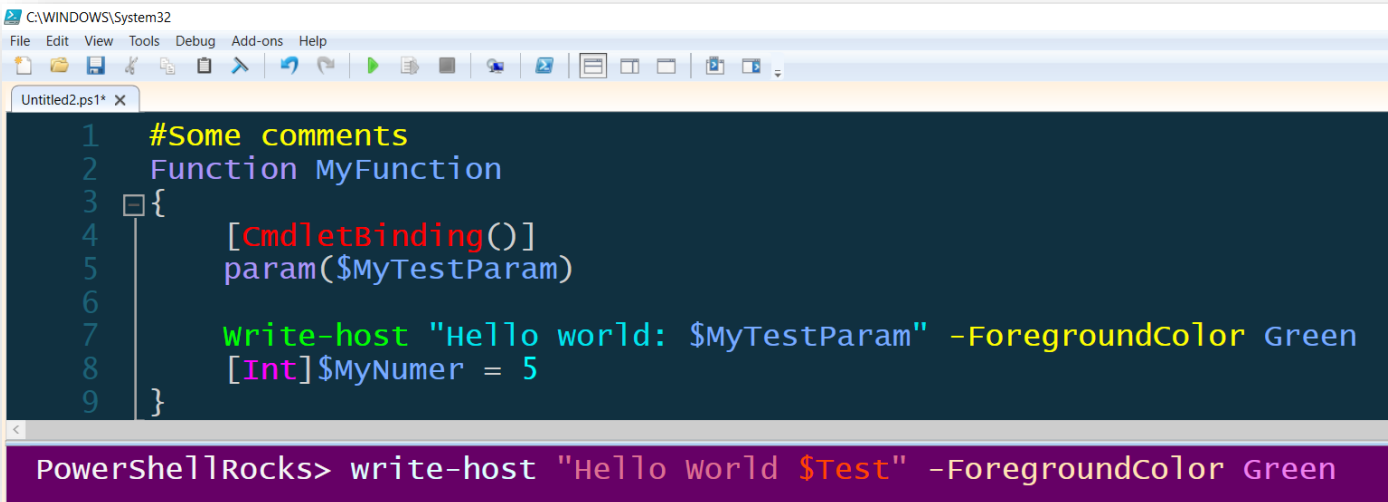
# Integrated Scripting Environment (ISE) - Features



# Syntax Color Highlighting

Color highlighting is automatic and **customizable**

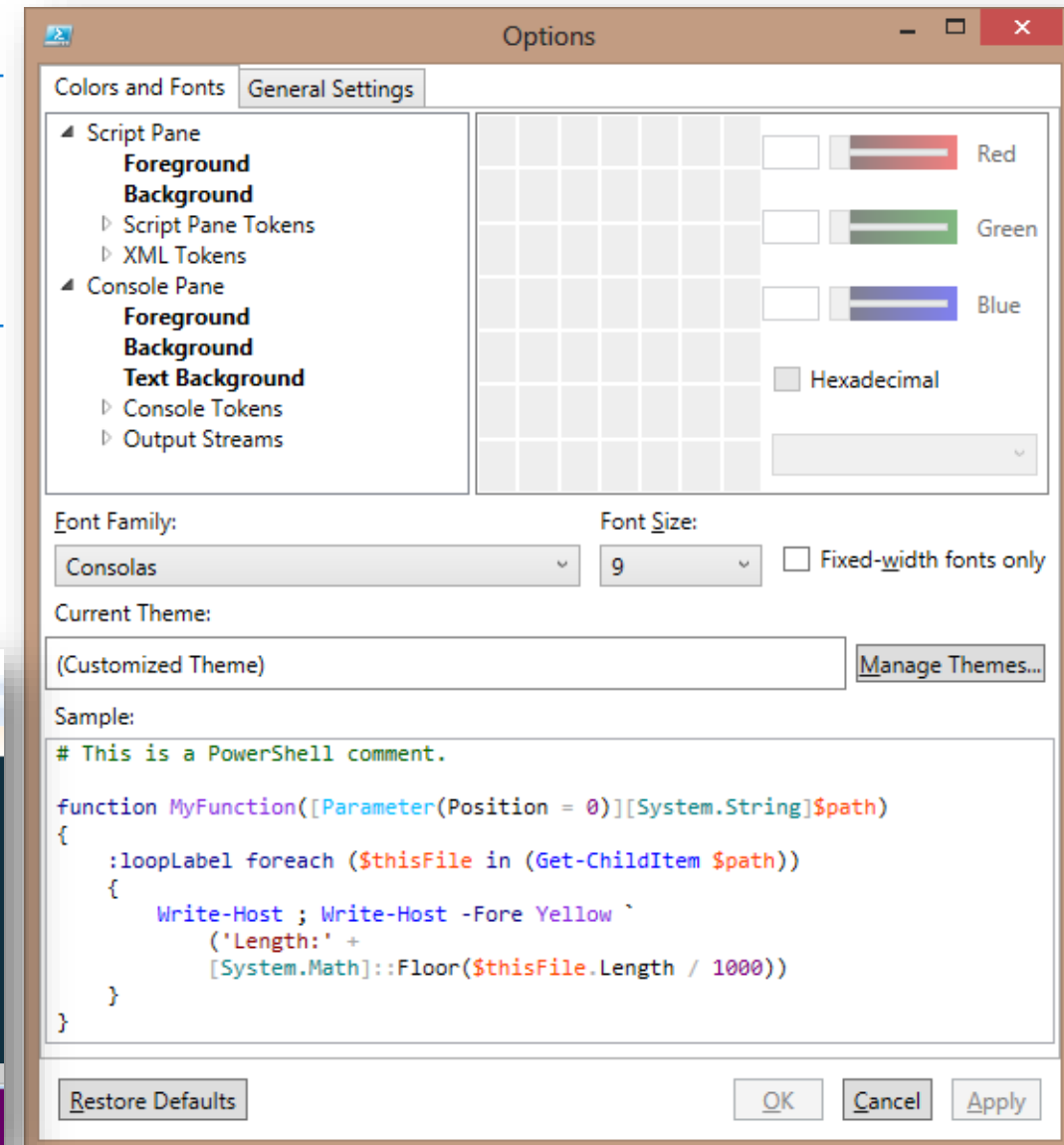
**Themes** to make common color sets easy to use



A screenshot of a PowerShell script editor window titled 'Untitled2.ps1\*'. The script content is as follows:

```
1 #Some comments
2 Function MyFunction
3 {
4     [CmdletBinding()]
5     param($MyTestParam)
6
7     write-host "Hello world: $MyTestParam" -ForegroundColor Green
8     [Int]$MyNumber = 5
9 }
```

The script is being executed in a terminal window at the bottom, showing the command: `PowerShellRocks> write-host "Hello world $Test" -ForegroundColor Green`. The terminal output is not visible.



# IntelliSense

---

Automatic graphic code completion

---

Force with Ctrl+Space

---

Cmdlet names

---

Parameter names

---

Parameter values

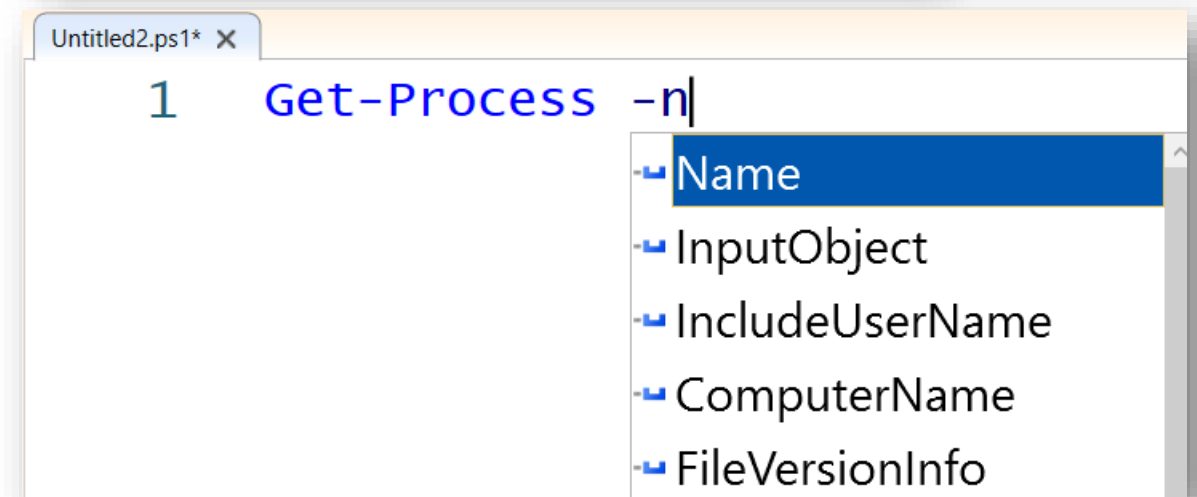
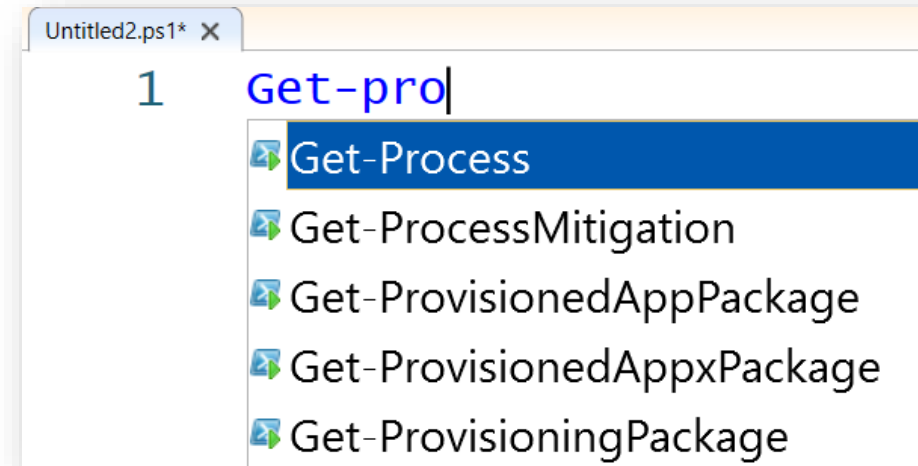
---

Object members

---

Scripting and console panes

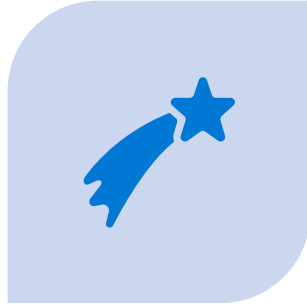
---



# Comments



Used mainly for leaving **notes** in a script for organization



Some **special** comments supported will be seen during the course



**Single** lines, and **blocks** of lines supported

```
#Single line comments
Write-Host "Hello world" #Inline comment

<#
  Block comment
  over multiple lines
#>
```

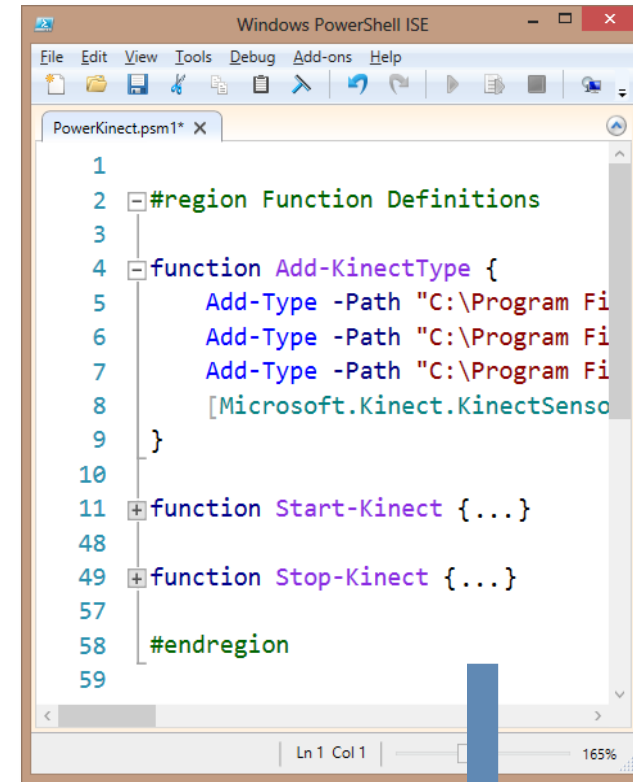
# Collapsible Code

**Visually minimize** sections of code

Matching **pairs** of parentheses, quotes, curly braces, etc.

Manual code regions for **organization** using special **comments**:

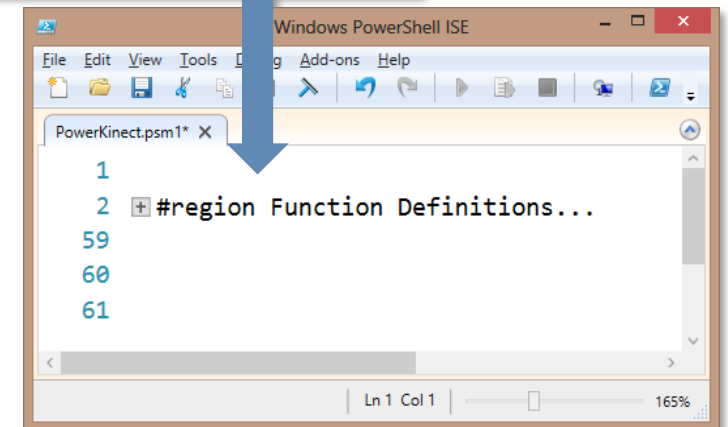
**#region** – Begin a Region  
**#endregion** – End a Region



A screenshot of the Windows PowerShell ISE window. The file is named 'PowerKinect.psm1'. The code is as follows:

```
1
2 #region Function Definitions
3
4 function Add-KinectType {
5     Add-Type -Path "C:\Program Fi
6     Add-Type -Path "C:\Program Fi
7     Add-Type -Path "C:\Program Fi
8     [Microsoft.Kinect.KinectSens
9 }
10
11 function Start-Kinect {...}
12
13 function Stop-Kinect {...}
14
15 #endregion
16
```

The code is color-coded: comments are green, function names are purple, and code inside functions is red. The 'Add-Type' commands are truncated. The status bar at the bottom shows 'Ln 1 Col 1' and '165%'.



A screenshot of the Windows PowerShell ISE window, showing the same file as the previous image. A blue arrow points from the first image to this one, indicating a transition. In this image, the first region is collapsed. The code is as follows:

```
1
2 + #region Function Definitions...
3
4
5
6
7
8
9
10
11
12
13
14
15
16
```

The status bar at the bottom shows 'Ln 1 Col 1' and '165%'.

# Additional Features

Compiled add-ons allow for rich functionality to be created, such as:

- Show command pane (built-in)
- variable watch window for debugging
- Spell checker

## Code Snippets

- Pre-written blocks of code
- Create custom ones with New-ISESnippet
- Accessible via Ctrl+J or Edit menu

## Auto-save and Crash Recovery

- Every 2 minutes keeps a backup of all open tabs to restore after closing

## Rich Copy

- Colors and font preserved when copying into word, outlook, etc

# Demonstration: ISE Features



