PowerShell Essential Training - Day 1

---

1. About the instructor

- **PowerShell Essential Training**
- **By:** Behrouz Amiri

---

2. What is PowerShell?

- **Definition:**
  - PowerShell is a task automation and configuration management framework from Microsoft, consisting of a command-line shell and associated scripting language.
- **Comparison with Programming Languages:**
  - **Scripting Languages:**
    - Lightweight
    - Designed for automating tasks
  - **Programming Languages:**
    - More complex
    - Used for building applications
- **PowerShell's Foundation:**
  - Built on top of the .NET framework, allowing access to .NET components.
- **Example:**

A simple PowerShell script to display "Hello, World":
powershell
Copy code

```
Write-Output "Hello, World"
```

  -
- **History:**
  - **PowerShell:**
    - Introduced in 2006 by Microsoft
    - Developed to improve system administration tasks
  - **.NET Framework:**
    - Introduced in 2002 by Microsoft
    - A software framework for building and running applications on Windows

---

3. Key Benefits for IT Professionals

- **Increased Productivity:**

    - Automate repetitive tasks
    - Integrate scripts into workflows
- **Error Reduction:**
    - Avoid manual errors through automation
- **Efficiency:**
    - Simplify complex tasks
- **Integration:**
    - Seamlessly integrates with other Microsoft tools
- **Example:**

Automating user account creation in Active Directory:
powershell
Copy code
```
New-ADUser -Name "John Doe" -SamAccountName "jdoe" -
UserPrincipalName "jdoe@domain.com"
```

    -

---

4. Installing and Configuring PowerShell

- **Versions:**
    - **Windows PowerShell 5.1:**
        - PowerShell.exe
    - **PowerShell Core 7.x:**
        - PWSH
- **Installation:**
    - Install on Windows, Linux, and macOS
    - Installation Guide
- **IDEs:**
    - **Visual Studio Code:**
        - Cross-platform, modern code editor
    - **Windows PowerShell ISE:**
        - Limited to v5.1
- **Example:**

Installing PowerShell Core on Windows:
powershell
Copy code
```
winget install --id Microsoft.Powershell --source winget
```

    -

---

5. File Types

- **PowerShell Scripts:**
  - `.ps1` files contain PowerShell commands and scripts.
- **PowerShell Modules:**
  - `.psm1` files contain reusable PowerShell functions and cmdlets.
- **PowerShell DataFiles:**
  - `.psd1` files are used for storing data in PowerShell.
- **Example:**

A simple PowerShell script file (`script.ps1`):
powershell
Copy code
```
Get-Process
```

  -

---

6. PowerShell Cmdlets

- **Anatomy:**
  - Verb-Noun –Parameter <ParameterValue>
  - Example: `Get-Process -Name "notepad"`
- **Approved Verbs:**
  - Approved Verbs for PowerShell Commands
- **Common Parameters:**
  - `WhatIf`, `Verbose`, `ErrorAction`
- **Example:**

Listing all running processes:
powershell
Copy code
```
Get-Process
```

  -

---

7. Using PowerShell Help

- **Get-Help Command:**
  - Example: `Get-Help Get-ADUser -Example`
- **Help Sources:**
  - Built-in help files
  - Online documentation
- **Updating Help Files:**
  - `Get-Help`
  - `Update-Help`

- **Example:**

Updating help files:
powershell
Copy code
```
Update-Help
```

    -

---

8. Running Cmdlets

- **Running Cmdlets:**
    - Example: `Get-Service`
    - Example: `Start-Service -Name "Spooler"`
- **Example:**

Stopping a service:
powershell
Copy code
```
Stop-Service -Name "Spooler"
```

    -

---

9. Understanding Object Output

- **Explanation:**
    - PowerShell commands output objects, not just text.
- **Example:**

`Get-Process` returns a list of process objects.
powershell
Copy code
```
Get-Process | Format-Table -Property Name, Id, CPU
```

    -

---

10. Variables and Data Types

- **Variables:**
    - Declaring: `$variableName = "Value"`
- **Data Types:**
    - **String:** `"Hello"`

    - **Integer:** `42`
    - **Array:** `@(1, 2, 3)`
- **Example:**

Using variables:
powershell
Copy code
```
$name = "John"
Write-Output "Hello, $name"
```

    -

---

11. Simple Scripting with PowerShell

- **Introduction to Scripting:**
    - Example script: Listing all running processes and stopping a specific process.
- **Example:**

A simple script (`stop_notepad.ps1`):
powershell
Copy code
```
Get-Process -Name "notepad" | Stop-Process
```

    -

---

12. Summary

- **Recap:**
    - Definition and benefits of PowerShell
    - Installation and configuration
    - File types and cmdlets
    - Help system and scripting basics

---

13. Q&A

- **Invitation for questions:**
    - Engage the audience