ISInc
Innovative Solutions

Windows PowerShell Scripting and Toolmaking

Course 55039A 5 Days Instructor-led, Hands on

Course Information

This five-day instructor-led course is intended for IT Professionals who have a working knowledge of Windows PowerShell 3.0 techniques and technologies, and who want to build reusable tools by using Windows PowerShell 3.0. Students of this course may administer a wide variety of server and client products and technologies that offer Windows PowerShell integration, including Microsoft Exchange Server, Microsoft Windows Active Directory Domain Services, Microsoft SharePoint Server, and more. This course focuses on the Windows PowerShell scripting language, and on the concepts and techniques needed to produce reusable, professional tools

This course is intended for administrators that have little or no programming experience, but who have a working knowledge of Windows PowerShell and who are able to use Windows PowerShell to run complex, interactive commands.

At Course Completion

After completing this course, students will be able to:

- Design tools, including input requirements, output requirements, and functional requirements.
- Write tools, including scripting, parameterizing commands, and providing verbose output.
- Debug tools and provide error handling within tools.
- Combine tools into script and manifest modules.
- Create custom formatting views.
- Create tools that are consistent in naming and operation with native Windows PowerShell tools

Prerequisites

Before attending this course, students must have:

- Experience in administering Windows server and client computers
- Experience in running interactive Windows PowerShell commands from the command prompt
- Course 10961 is strongly recommended as a pre-requisite to this course

ISInc
Innovative Solutions

Course Outline

Module 1: Preparing for Scripting

This module explains how to prepare the environment for scripting, and provides refresher and background information for scripting

Lessons

- Securing the Scripting Environment
- Understanding Variables and Operators
- Understanding Scripting Constructs and Scope

After completing this module, students will be able to:

- Describe and set the execution policy.
- Run Windows PowerShell scripts.
- Use variables and operators.
- Describe and use scripting constructs.
- Describe the operation of Windows PowerShell scope.

Module 2: Parameterizing a Command

This module explains how to start with an existing command and parameterize it to create a reusable tool.

Lessons

- Designing Parameters
- Implementing Parameters

Lab : Preparing a Command

- Identify changeable values
- Declare parameters
- Use parameters in place of changeable values
- Test the script

After completing this module, students will be able to:

- Evolve a command into a parameterized script

Module 3: Creating a Script Module

This module explains how to turn a basic script into a script module that can be distributed, loaded, and unloaded in Windows PowerShell.

Lessons

- Designing Script Modules
- Implementing Script Modules

Lab : Creating a Script Module

- Creating a Script Module
- Saving the script module
- Adding a module-level variable
- Controlling module member visibility
- Testing the script module

After completing this module, students will be able to:

- Create a script module based upon an existing script or function.

Module 4: Handling Errors

This module explains how to trap and handle errors within a script module.

Lessons

- Designing Error Handling
- Implementing Error Handling

Lab : Handling Errors

- Using the Try…Catch Construct
- Handling Command Errors
- Handling Non-Command Errors
- Logging Errors to a File
- Displaying Warning Messages

After completing this module, students will be able to:

- Describe and use the Try…Catch construct.
- Handle command errors.
- Handle non-command errors.
- Log errors to a file.
- Display warning messages.

Module 5: Writing Commands that Use Pipeline Input and Output

This module explains how to write commands that integrate with the Windows PowerShell pipeline. Students will create commands that produce pipeline output and that accept pipeline input.

Lessons

- Understanding Pipeline Parameter Binding
- Implementing Pipeline Parameter Input
- Implementing Pipeline Parameter Input

Lab : Writing Commands that Use Pipeline Inout and Output

- Adding Pipeline Input Capability to Parameters
- Working with Pipeline Input
- Creating Custom Output Objects
- Outputting Objects to the Pipeline

After completing this module, students will be able to:

- Create commands that accept pipeline input.
- Create commands that consolidate multiple data sources into Windows PowerShell pipeline output.
- Describe how location discovery works.

Module 6: Creating Hierarchical Command Output

This module explains how to create, and use, object-oriented output that includes object hierarchies.

Lessons

- Designing Complex Command Output
- Implementing Complex Command Output
- Using Object Hierarchies

Lab: Creating Hierarchical Command Output

- Retrieving and Enumerating Data
- Creating Child Objects
- Creating the Parent Object
- Displaying and Object Hierarchy
- Persisting an Object Hierarchy

After completing this module, students will be able to:

- Create hierarchical, object-oriented command output.
- Use hierarchical, object—oriented command output.

Module 7: Debugging Scripts

This module explains Windows PowerShell techniques used to debug scripts, and provides students with opportunities to practice debugging skills.

Lessons

- Designing Scripts for Debugging
- Implementing Script Debugging

Lab : Debugging Scripts

- Using Write-Debug
- Using PSBreakpoints

After completing this module, students will be able to:

- Debug scripts by using Write-Debug.
- Debug scripts by using PSBreakpoints.

Module 8: Customizing Default Formatting

This module explain how to create a custom formatting view that can be added to a script module.

Lessons

- Designing Formatting
- Implementing Custom Formatting

Lab: Customizing Default Formatting

- Adding a Custom Type Name to an Object
- Creating a DefaultDisplayPropertySet Type Extension
- Creating a Custom View
- Adding Type Extensions and Views to Modules and Creating a Module Manifest

After completing this module, students will be able to:

- Create custom type extensions.
- Create custom views.

Module 9: Adding Advanced Parameter Attributes and Command Documentation

This module explains how to declare parameter aliases, help messages, and input validation. It also explains how to implement switch parameters, how to add support for the –WhatIf and –Confirm parameters, and how to add comment-based help to a command.

Lessons

- Implementing Advanced Parameter Attributes
- Implementing Help Documentation

Lab : Adding Advanced Parameter Attributes and Command Documentation

- Defining Aliases and Help Messages
- Defining Parameter Validation
- Adding Comment-Based Help
- Writing a Command that Uses –Confirm and –WhatIf

After completing this module, students will be able to:

- Add advanced parameter attributes, including aliases and validation.
- Create comment-based documentation for commands.
- Write commands that use –WhatIf and –Confirm parameters.

Module 10: Creating Controller Scripts

This module explains how to create scripts that implement complex business processes by running multiple tools in a specified sequence

Lessons

- Designing Script Execution
- Implementing a Controller Script

Lab : Creating Controller Scripts

- Creating a Controller Script
- Parameterizing a Controller Script
- Testing a Controller Script
- Debugging a Controller Script

After completing this module, students will be able to:

- Implement controller scripts by combining specified tools.
- Test and debug controller scripts.

Module 11: Creating HTML-Based Reports

This module explains how to write controller scripts that produce HTML-based management reports.

Lessons

- Creating Basic HTML Reports
- Creating Enhanced HTML Reports

Lab : Creating Reports by using HTML

- Creating Reports by using HTML
- Converting Objects into HTML Fragments
- Combining HTML Fragments
- Adding Basic Formatting
- Creating Enhanced HTML Fragments

- Applying Conditional Formatting

After completing this module, students will be able to:

- Create basic and enhanced HTML reports that include specified management information.

Module 12: Creating Basic Workflows

This modules explains the key differences between Windows PowerShell functions and workflows, and shows students how to create a basic workflow.

Lessons

- Understanding Workflows
- Implementing Workflows

Lab : Creating Basic Workflows

- Importing the PSWorkflow Module
- Converting a Function to a Basic Worklfow
- Parallelizing Commands

After completing this module, students will be able to:

- Describe the differences between a Windows PowerShell function and a workflow
- Convert a function to a workflow
- Run a workflow that includes parallel execution

Module 13: Working with XML Data

This module explains how Windows PowerShell interprets, represents, and manipulates XML-based data.

Lessons

- Understanding XML
- Implementing XML Manipulation

Lab : Working with XML Data

- Loading XML
- Manipulating XML as an Object Hierarchy
- Selecting XML Elements by using XPath
- Modifying XML
- Saving XML

After completing this module, students will be able to:

- Load, manipulate, and save data in XML formats.

Module 14: Using Advanced Scripting Techniques

This module explains how to use advanced scripting techniques, including execution of external commands and graphical user interfaces.

Lessons

- Using External Functionality
- Adding Graphical User Interface Elements

After completing this module, students will be able to:

- Run external commands from inside Windows PowerShell
- Describe the process required to create a graphical user interface in Windows PowerShell

Module 15: Creating Proxy Functions

This module explains how to create proxy functions in Windows PowerShell.

Lessons

- Designing Proxy Functions
- Implementing Proxy Functions

Lab : Creating Proxy Functions

- Generating a Proxy Function Template
- Modifying the Template
- Using the Proxy Function
- Bypassing a Proxy Function

After completing this module, students will be able to:

- Create and modify proxy functions in Windows PowerShell

Module 16: Building Tools in Windows PowerShell

This module is a "final exam" for the course, and offers students the opportunity to build a complete tool, from scratch, using many of the techniques that they have learned in the preceding days.

Lessons

- Designing the Tool
- Implementing the Tool
- Testing the Tool

Lab : Building Tools in Windows PowerShell

- Designing the Tool
- Implementing the Tool
- Testing the Tool

After completing this module, students will be able to:

- Design, create, and test tools in Windows PowerShell