

Course duration

- 5 days

Course Benefits

- Learn to describe the correct patterns for building modularized tools in Windows PowerShell.
- Learn to build highly modularized functions that comply with native PowerShell patterns.
- Learn to build controller scripts that expose user interfaces and automate business processes.
- Learn to manage data in a variety of formats.
- Learn to write automated tests for tools.
- Learn to debug tools.

Available Delivery Methods

Public Class

Public expert-led online training from the convenience of your home, office or anywhere with an internet connection. Guaranteed to run .

Private Class

Private classes are delivered for groups at your offices or a location of your choice.

Course Outline

1. Tool Design
 1. Tools do one thing
 2. Tools are flexible
 3. Tools look native
 4. Lab 1: Designing a Tool
 5. Design a tool
2. Start with a Command
 1. Why start with a command?
 2. Discovery and experimentation
 3. Lab 1: Designing a Tool
 4. Start with a command
3. Build a Basic Function and Module
 1. Start with a basic function

2. Create a script module
3. Check prerequisites
4. Run the new command
5. Lab 1: Designing a Tool
6. Build a basic function and module
4. Adding CmdletBinding and Parameterizing
 1. About CmdletBinding and common parameters
 2. Accepting pipeline input
 3. Mandatory-ness
 4. Parameter validation
 5. Parameter aliases
 6. Lab 1: Designing a Tool
 7. Adding CmdletBinding and Parameterizing
5. Emitting Objects as Output
 1. Assembling information
 2. Constructing and emitting output
 3. Quick tests
 4. Lab 1: Designing a Tool
 5. Emitting objects as output
6. An Interlude: Changing Your Approach
 1. Examining a script
 2. Critiquing a script
 3. Revising the script
7. Using Verbose, Warning, and Informational Output
 1. Knowing the six channels
 2. Adding verbose and warning output
 3. Doing more with verbose output
 4. Informational output
 5. Lab 1: Designing a Tool
 6. Using Verbose, Warning, and Informational Output
8. Comment-Based Help
 1. Where to put your help
 2. Getting started
 3. Going further with comment-based help
 4. Broken help
 5. Lab 1: Designing a Tool
 6. Comment-based help
9. Handling Errors
 1. Understanding errors and exceptions
 2. Bad handling
 3. Two reasons for exception handling
 4. Handling exceptions in our tool
 5. Capturing the actual exception
 6. Handling exceptions for non-commands
 7. Going further with exception handling
 8. Deprecated exception handling
 9. Lab 1: Designing a Tool

- 10. Handling errors
- 10. Basic Debugging
 - 1. Two kinds of bugs
 - 2. The ultimate goal of debugging
 - 3. Developing assumptions
 - 4. Write-Debug
 - 5. Set-PSBreakpoint
 - 6. The PowerShell ISE
 - 7. Lab 1: Designing a Tool
 - 8. Basic debugging
- 11. Going Deeper with Parameters
 - 1. Parameter positions
 - 2. Validation
 - 3. Multiple parameter sets
 - 4. Value from remaining arguments
 - 5. Help messages
 - 6. Aliases
 - 7. More CmdletBinding
- 12. Writing Full Help
 - 1. External help
 - 2. Using PlatyPs
 - 3. Supporting online help
 - 4. “About” topics
 - 5. Making your help updatable
 - 6. Lab 1: Designing a Tool
 - 7. Writing full help
- 13. Unit Testing Your Code
 - 1. Sketching out the test
 - 2. Making something to test
 - 3. Expanding the test
 - 4. Going further with Pester
 - 5. Lab 1: Designing a Tool
 - 6. Unit testing your code
- 14. Extending Output Types
 - 1. Understanding types
 - 2. The Extensible Type System
 - 3. Extending an object
 - 4. Using Update-TypeData
- 15. Analyzing Your Script
 - 1. Performing a basic analysis
 - 2. Analyzing the analysis
 - 3. Lab 1: Designing a Tool
 - 4. Analyzing your script
- 16. Publishing Your Tools
 - 1. Begin with a manifest
 - 2. Publishing to PowerShell Gallery
 - 3. Publishing to private repositories

4. Lab 1: Designing a Tool
 5. Publishing your tools
17. Basic Controllers: Automation Scripts and Menus
 1. Building a menu
 2. Using UIChoice
 3. Writing a process controller
 4. Lab 1: Designing a Tool
 5. Basic controllers
18. Proxy Functions
 1. A proxy example
 2. Creating the proxy base
 3. Modifying the proxy
 4. Adding or removing parameters
 5. Lab 1: Designing a Tool
 6. Proxy functions
19. Working with XML Data
 1. Simple: CliXML
 2. Importing native XML
 3. ConvertTo-XML
 4. Creating native XML from scratch
 5. Lab 1: Designing a Tool
 6. Working with XML
20. Working with JSON Data
 1. Converting to JSON
 2. Converting from JSON
 3. Lab 1: Designing a Tool
 4. Working with JSON data
21. Working with SQL Server Data
 1. SQL Server terminology and facts
 2. Connecting to the server and database
 3. Writing a query
 4. Running a query
 5. Invoke-SqlCmd
 6. Thinking about tool design patterns
22. Final Exam
 1. Lab problem
 2. Break down the problem
 3. Do the design
 4. Test the commands
 5. Code the tool

Class Materials

Each student will receive a comprehensive set of materials, including course notes and all the

class examples.

Class Prerequisites

Experience in the following *is required* for this PowerShell class:

- Experience in administering Windows server and client computers.
- Experience in running interactive Windows PowerShell commands from the command prompt.
- MOC0961 is strongly recommended as a prerequisite to this course.

Prerequisite Courses

Courses that can help you meet these prerequisites:

- [MOC 10961B - Automating Administration with Windows PowerShell](#)