Administering Microsoft Endpoint Configuration Manager (M20703)

Duration: 5 Days **Course Code: M55348**

Overview:

This three- to five-day instructor-led is intended for IT professionals who are interested in furthering their skills in Windows PowerShell and administrative automation. The course assumes a basic working knowledge of PowerShell as an interactive command-line shell, and teaches students the correct patterns and practices for building reusable, tightly scoped units of automation.

Virtual Learning

This interactive training can be taken from any location, your office or home and is delivered by a trainer. This training does not have any delegates in the class with the instructor, since all delegates are virtually connected. Virtual delegates do not travel to this course, Global Knowledge will send you all the information needed before the start of the course and you can test the logins.

Target Audience:

This course is intended for administrators in a Microsoft-centric environment who want to build reusable units of automation, automate business processes, and enable less-technical colleagues to accomplish administrative tasks.

Objectives:

- **At Course Completion**
- Describe the correct patterns for building modularized tools in Windows PowerShell
- Build highly modularized functions that comply with native PowerShell patterns
- Build controller scripts that expose user interfaces and automate business processes
- Manage data in a variety of formats
- Write automated tests for tools
- Debug tools

Prerequisites:

- Experience at basic Windows administration
- Experience using Windows PowerShell to query and modify system information
- Experience using Windows PowerShell to discover commands and their usage
- Experience using WMI and/or CIM to query system information

Content:

Module 1: Tool Design

This module explains how to design tools and units of automation that comply with native PowerShell usage patterns.

Lessons M1

- Tools do one thing
- Tools are flexible
- Tools look native

Lab 1: Designing a Tool

- Design a tool
- Start with a command
- Build a basic function and module
- Adding CmdletBinding and Parameterizing
- Emitting objects as output
- Using Verbose, Warning, and Informational Output
- Comment-based help
- Handling errors
- Basic debugging
- Writing full help
- Unit testing your code
- Analyzing your script
- Publishing your tools
- Basic controllers
- Proxy functions
- Working with XML
- Working with JSON data

After completing module 1, students will be able to:

- Describe the native shell patterns that a good tool design should exhibit

Module 2: Start with a Command

This module explains how to start the scripting process by beginning in the interactive shell console.

Lessons M2

- Why start with a command?
- Discovery and experimentation

Lab 1: Designing a Tool

- Design a tool
- Start with a command
- Build a basic function and module
- Adding CmdletBinding and Parameterizing
- Emitting objects as output
- Using Verbose, Warning, and Informational Output

Lessons M8

- Where to put your help
- Getting started
- Going further with comment-based help
- Broken help

Lab 1: Designing a Tool

- Design a tool
- Start with a command
- Build a basic function and module
- Adding CmdletBinding and Parameterizing
- Emitting objects as output
- Using Verbose, Warning, and Informational Output
- Comment-based help
- Handling errors
- Basic debugging
- Writing full help
- Unit testing your code
- Analyzing your script
- Publishing your tools
- Basic controllers
- Proxy functions
- Working with XML
- Working with JSON data

After completing module 8, students will be able to:

- Describe the purpose and construction of comment-based help
- Add comment-based help to a function
- Identify causes of broken comment-based help

Module 9: Handling Errors

This module explains how to create tools that deal with anticipated errors.

Lessons M9

- Understanding errors and exceptions
- Bad handling
- Two reasons for exception handling
- Handling exceptions in our tool
- Capturing the actual exception
- Handling exceptions for non-commands
- Going further with exception handling
- Deprecated exception handling

Lab 1: Designing a Tool

- Design a tool
- Start with a command
- Build a basic function and module
- Adding CmdletBinding and Parameterizing

After completing module 15, students will be able to:

- Describe the use of Script Analyzer
- Perform a basic script analysis

Module 16: Publishing Your Tools

This module explains how to publish tools to public and private repositories.

Lessons 16

- Begin with a manifest
- Publishing to PowerShell Gallery
- Publishing to private repositories

Lab 1: Designing a Tool

- Design a tool
- Start with a command
- Build a basic function and module
- Adding CmdletBinding and Parameterizing
- Emitting objects as output
- Using Verbose, Warning, and Informational Output
- Comment-based help
- Handling errors
- Basic debugging
- Writing full help
- Unit testing your code
- Analyzing your script
- Publishing your tools
- Basic controllers
- Proxy functions
- Working with XML
- Working with JSON data

After completing module 16, students will be able to:

- Describe the tool publishing process and requirements
- Publish a tool to a repository

Module 17: Basic Controllers: Automation Scripts and Menus

This module explains how to create controller scripts that put tools to use.

Lessons 17

- Building a menu
- Using UIChoice
- Writing a process controller

Lab 1: Designing a Tool

- Comment-based help
- Handling errors
- Basic debugging
- Writing full help
- Unit testing your code
- Analyzing your script
- Publishing your tools
- Basic controllers
- Proxy functions
- Working with XML
- Working with JSON data

After completing module 2, students will be able to:

- Describe the benefits of discovery and experimentation in the console
- Discover and experiment with existing commands in the console

Module 3: Build a Basic Function and Module

This module explains how to build a basic function and module, using commands already experimented with in the shell.

Lessons M3

- Start with a basic function
- Create a script module
- Check prerequisites
- Run the new command

Lab 1: Designing a Tool

- Design a tool
- Start with a command
- Build a basic function and module
- Adding CmdletBinding and Parameterizing
- Emitting objects as output
- Using Verbose, Warning, and Informational Output
- Comment-based help
- Handling errors
- Basic debugging
- Writing full help
- Unit testing your code
- Analyzing your script
- Publishing your tools
- Basic controllers
- Proxy functions
- Working with XML
- Working with JSON data

After completing module 3, students will be able to:

- Build a basic function
- Create a script module
- Run a command from a script module

Module 4: Adding CmdletBinding and Parameterizing

- Emitting objects as output
- Using Verbose, Warning, and Informational Output
- Comment-based help
- Handling errors
- Basic debugging
- Writing full help
- Unit testing your code
- Analyzing your script
- Publishing your tools
- Basic controllers
- Proxy functions
- Working with XML
- Working with JSON data

After completing module 9, students will be able to:

- Describe the native patterns for handling errors in a command
- Add error handling to a command
- Run a command and observe error handling behaviors

Module 10: Basic Debugging

This module explains how to use native PowerShell script debugging tools.

Lessons M10

- Two kinds of bugs
- The ultimate goal of debugging
- Developing assumptions
- Write-Debug
- Set-PSBreakpoint
- The PowerShell ISE

Lab 1: Designing a Tool

- Design a tool
- Start with a command
- Build a basic function and module
- Adding CmdletBinding and Parameterizing
- Emitting objects as output
- Using Verbose, Warning, and Informational Output
- Comment-based help
- Handling errors
- Basic debugging
- Writing full help
- Unit testing your code
- Analyzing your script
- Publishing your tools
- Basic controllers
- Proxy functions
- Working with XML
- Working with JSON data

After completing module 10, students will be able to:

- Design a tool
- Start with a command
- Build a basic function and module
- Adding CmdletBinding and Parameterizing
- Emitting objects as output
- Using Verbose, Warning, and Informational Output
- Comment-based help
- Handling errors
- Basic debugging
- Writing full help
- Unit testing your code
- Analyzing your script
- Publishing your tools
- Basic controllers
- Proxy functions
- Working with XML
- Working with JSON data

After completing module 17, students will be able to:

- Describe the purpose of basic controller scripts
- Write a simple controller script

Module 18: Proxy Functions

This module explains how to create and use proxy functions.

Lessons 18

- A proxy example
- Creating the proxy base
- Modifying the proxy
- Adding or removing parameters

Lab 1: Designing a Tool

- Design a tool
- Start with a command
- Build a basic function and module
- Adding CmdletBinding and Parameterizing
- Emitting objects as output
- Using Verbose, Warning, and Informational Output
- Comment-based help
- Handling errors
- Basic debugging
- Writing full help
- Unit testing your code
- Analyzing your script
- Publishing your tools
- Basic controllers
- Proxy functions
- Working with XML
- Working with JSON data

After completing module 18, students will be able to:

- Describe the purpose of proxy functions

This module explains how to extend the functionality of a tool, parameterize input values, and use CmdletBinding.

Lessons M4

- About CmdletBinding and common parameters
- Accepting pipeline input
- Mandatory-ness
- Parameter validation
- Parmeter aliases

Lab 1: Designing a Tool

- Design a tool
- Start with a command
- Build a basic function and module
- Adding CmdletBinding and Parameterizing
- Emitting objects as output
- Using Verbose, Warning, and Informational Output
- Comment-based help
- Handling errors
- Basic debugging
- Writing full help
- Unit testing your code
- Analyzing your script
- Publishing your tools
- Basic controllers
- Proxy functions
- Working with XML
- Working with JSON data

After completing module 4, students will be able to:

- Describe the purpose of CmdletBinding and list common parameters
- Parameterize a script's input
- Define parameters as mandatory
- Define parameters as accepting pipeline input
- Define parameter validation

Module 5: Emitting Objects as Output

This module explains how to create tools that produce custom objects as output.

Lessons M5

- Assembling information
- Constructing and emitting output
- Quick tests

Lab 1: Designing a Tool

- Design a tool
- Start with a command
- Describe the tools used for debugging in PowerShell
- Debug a broken script

Module 11: Going Deeper with Parameters

This module explains how to further define parameter attributes in a PowerShell command.

Lessons M11

- Parameter positions
- Validation
- Multiple parameter sets
- Value from remaining arguments
- Help messages
- Aliases
- More CmdletBinding

Lab 1: No Lab

- Click here to enter text.
- Click here to enter text.
- Click here to enter text.
- Click here to enter text.

After completing module 11, students will be able to:

- Describe the use of positional parameters
- Describe additional parameter validation methods
- Describe how to define multiple parameter sets
- Describe other parameter definition options

Module 12: Writing Full Help

This module explains how to create external help for a command.

Lessons M12

- External help
- Using PlatyPs
- Supporting online help
- "About" topics
- Making your help updatable

Lab 1: Designing a Tool

- Design a tool
- Start with a command
- Build a basic function and module
- Adding CmdletBinding and Parameterizing
- Emitting objects as output
- Using Verbose, Warning, and Informational Output
- Create a simple proxy function

Module 19: Working with XML Data

This module explains how to work with XML data in PowerShell.

Lessons 19

- Simple: CliXML
- Importing native XML
- ConvertTo-XML
- Creating native XML from scratch

Lab 1: Designing a Tool

- Design a tool
- Start with a command
- Build a basic function and module
- Adding CmdletBinding and Parameterizing
- Emitting objects as output
- Using Verbose, Warning, and Informational Output
- Comment-based help
- Handling errors
- Basic debugging
- Writing full help
- Unit testing your code
- Analyzing your script
- Publishing your tools
- Basic controllers
- Proxy functions
- Working with XML
- Working with JSON data

After completing module 19, students will be able to:

- Describe the use of XML within PowerShell
- Use XML data within a PowerShell function

Module 20: Working with JSON Data

This module explains how to using JSON data in PowerShell.

Lessons 20

- Converting to JSON
- Converting from JSON

Lab 1: Designing a Tool

- Design a tool
- Start with a command
- Build a basic function and module
- Adding CmdletBinding and Parameterizing
- Emitting objects as output
- Using Verbose, Warning, and Informational Output
- Comment-based help

- Build a basic function and module
- Adding CmdletBinding and Parameterizing
- Emitting objects as output
- Using Verbose, Warning, and Informational Output
- Comment-based help
- Handling errors
- Basic debugging
- Writing full help
- Unit testing your code
- Analyzing your script
- Publishing your tools
- Basic controllers
- Proxy functions
- Working with XML
- Working with JSON data

After completing module 5, students will be able to:

- Describe the purpose of object-based output
- Create and output custom objects from a function

Module 6: An Interlude: Changing Your Approach

This module explains how to re-think tool design, using concrete examples of how it's often done wrong.

Lessons M6

- Examining a script
- Critiquing a script
- Revising the script

Lab 1: No lab

- Click here to enter text.
- Click here to enter text.
- Click here to enter text.
- Click here to enter text.

After completing module 6, students will be able to:

- Describe the native patterns that a good tool design should exhibit
- Redesign a script to meet business requirements and conform to native patterns

Module 7: Using Verbose, Warning, and Informational Output

This module explains how to use additional output pipelines for better script behaviors.

Lessons M7

- Knowing the six channels
- Comment-based help
- Handling errors
- Basic debugging
- Writing full help
- Unit testing your code
- Analyzing your script
- Publishing your tools
- Basic controllers
- Proxy functions
- Working with XML
- Working with JSON data

After completing module 12, students will be able to:

- Describe the advantages of external help
- Create external help using PlatyPS and Markdown

Module 13: Unit Testing Your Code

This module explains how to use Pester to perform basic unit testing.

Lessons M13

- Sketching out the test
- Making something to test
- Expanding the test
- Going further with Pester

Lab 1: Designing a Tool

- Design a tool
- Start with a command
- Build a basic function and module
- Adding CmdletBinding and Parameterizing
- Emitting objects as output
- Using Verbose, Warning, and Informational Output
- Comment-based help
- Handling errors
- Basic debugging
- Writing full help
- Unit testing your code
- Analyzing your script
- Publishing your tools
- Basic controllers
- Proxy functions
- Working with XML
- Working with JSON data

After completing module 13, students will be able to:

- Describe the purpose of unit testing
- Write basic unit tests for PowerShell functions

Module 14: Extending Output Types

- Handling errors
- Basic debugging
- Writing full help
- Unit testing your code
- Analyzing your script
- Publishing your tools
- Basic controllers
- Proxy functions
- Working with XML
- Working with JSON data

After completing module 20, students will be able to:

- Describe the use of JSON data within PowerShell
- Use JSON data within a PowerShell function

Module 21: Working with SQL Server Data

This module explains how to use SQL Server from within a PowerShell script.

Lessons 21

- SQL Server terminology and facts
- Connecting to the server and database
- Writing a query
- Running a query
- Invoke-SqlCmd
- Thinking about tool design patterns

Lab 1: No Lab

- Click here to enter text.
- Click here to enter text.
- Click here to enter text.
- Click here to enter text.

After completing module 21, students will be able to:

- Describe the use of SQL Server from within PowerShell
- Write and run SQL Server queries
- Design tools that use SQL Server for data storage

Module 22: Final Exam

This module provides a chance for students to use everything they have learned in this course within a practical example.

Lessons 22

- Lab problem
- Break down the problem
- Do the design
- Test the commands

- Adding verbose and warning output
- Doing more with verbose output
- Informational output

Lab 1: Designing a Tool

- Design a tool
- Start with a command
- Build a basic function and module
- Adding CmdletBinding and Parameterizing
- Emitting objects as output
- Using Verbose, Warning, and Informational Output
- Comment-based help
- Handling errors
- Basic debugging
- Writing full help
- Unit testing your code
- Analyzing your script
- Publishing your tools
- Basic controllers
- Proxy functions
- Working with XML
- Working with JSON data

After completing module 7, students will be able to:

- Describe the six output channels in the shell
- Write commands that use verbose, warning, and informational output
- Run commands with extra output enabled

Module 8: Comment-Based Help

This module explains how to add comment-based help to tools.

This module explains how to extend objects with additional capabilities.

Lessons M14

- Understanding types
- The Extensible Type System
- Extending an object
- Using Update-TypeData

Lab 1: No Lab

- Click here to enter text.
- Click here to enter text.
- Click here to enter text.
- Click here to enter text.

After completing module 14, students will be able to:

- Describe the purpose of the ETS
- Extend an existing object type

Module 15: Analyzing Your Script

This module explains how to use Script Analyzer to support best practices and prevent common problems.

Lessons 15

- Performing a basic analysis
- Analyzing the analysis

Lab 1: Designing a Tool

- Design a tool
- Start with a command
- Build a basic function and module
- Adding CmdletBinding and Parameterizing
- Emitting objects as output
- Using Verbose, Warning, and Informational Output
- Comment-based help
- Handling errors
- Basic debugging
- Writing full help
- Unit testing your code
- Analyzing your script
- Publishing your tools
- Basic controllers
- Proxy functions
- Working with XML
- Working with JSON data

- Code the tool

Lab 1: Final Exam

- Lab one

Lab 2: Final Exam

- Lab two

After completing module 22, students will be able to:

- Create PowerShell tools, using native design patterns, from business requirements.

Further Information:

For More information, or to book your course, please call us on Head Office Tel.: +974 40316639

training@globalknowledge.qa

www.globalknowledge.com/en-qa/

Global Knowledge, Qatar Financial Center, Burj Doha,Level 21, P.O.Box 27110, West Bay, Doha, Qatar