Microsoft

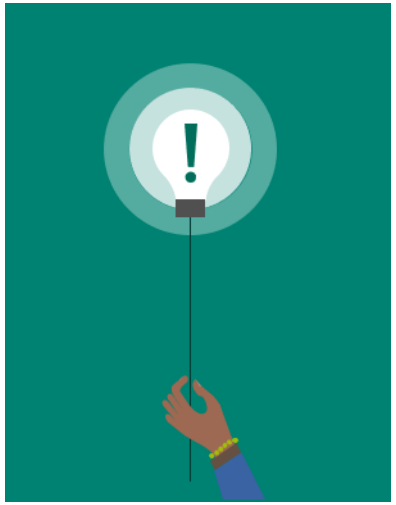
Debugging in PowerShell

Module 5

Objectives

Learnings covered:

- ✓Introduction to Debugging
- ✓Syntax Errors
- ✓Break Points
- ✓Step over / Step into / Step out



Introduction to Debugging

Syntax error examples

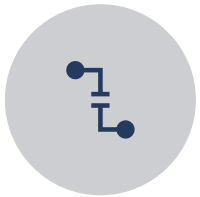
```
function foo {
    Write-Output "Function foo'
}
```

```
function foo {
    Write-Output "Function foo"
```

```
function foo {
if ($true) {
Write-Output 'true'
} }
```

Understanding BreakPoints

Line BreakPoint



Command
BreakPoint

Variable
BreakPoint

Line BreakPoint

Debugging Task	Description	How to accomplish it in PowerShell ISE
Toggle BreakPoint	A breakpoint can be set in a script only after it has been saved. Right-click the line where you want to set a line breakpoint, and then click **Toggle Breakpoint**. Or, click the line where you want to set a line breakpoint, and press F9 or, on the **Debug** menu, click **Toggle Breakpoint**.	**F9** **Powershell command:** Set-PSBreakpoint -Script .\script.ps1 -Line 8
Remove a breakpoint	Removing a breakpoint deletes it. Right-click the line where you want to remove a breakpoint, and then click ToggleBreakpoint.	**F9** **Powershell command:** Remove-PSBreakpoint –id 8
Start debugging	Press F5 or, on the toolbar, click the Run Script icon, or on the Debug menu click Run/Continue. The script runs until it encounters the first breakpoint. It pauses operation there and highlights the line on which it paused.	**F5**
Stop debugging	Stop the debugger and return to the shell.	Press **SHIFT+F5** or, on the **Debug** menu, click **Stop** Debugger, or, in the **Console** Pane, type **Q** and then press ENTER.

How to step over, step into, and step out while debugging

Debugging Task	Description	How to accomplish it in PowerShell ISE
Step Into	Executes the current statement and then stops at the next statement. If the current statement is a function or script call, then the debugger steps into that function or script, otherwise it stops at the next statement.	Press F11 or, on the **Debug** menu, click **Step Into**, or in the Console Pane, type S and press ENTER.
Step Over	Executes the current statement and then stops at the next statement. If the current statement is a function or script call, then the debugger executes the whole function or script, and it stops at the next statement after the function call.	Press F10 or, on the **Debug** menu, click **Step Over**, or in the Console Pane, type V and press ENTER.
Step Out	Steps out of the current function and up one level if the function is nested. If in the main body, the script is executed to the end, or to the next breakpoint. The skipped statements are executed, but not stepped through.	Press SHIFT+F11, or on the **Debug** menu, click **Step Out**, or in the Console Pane, type O and press ENTER.
Continue	Continues execution to the end, or to the next breakpoint. The skipped functions and invocations are executed, but not stepped through.	Press F5 or, on the **Debug** menu, click **Run/Continue**, or in the Console Pane, type C and press ENTER.

Variable / Command BreakPoint

- You can add breakpoints to a variable on read/write/readwrite actions.

Add a variable breakpoint:

```
Set-PSBreakpoint -Script .\do-something.ps1 -Variable file -Mode ReadWrite
-Action { "The value is $file" | Out-File -FilePath c:\temp\2.txt -Append}
```

Add a command breakpoint:

```
Set-PSBreakpoint -Script .\do-something.ps1 -Command 'Write-Output’
-Action { 'line' | Out-File -FilePath 'c:\temp\1.txt' -Append}
```

Demonstration

Breakpoints

Step over / Step into / Step out

Questions?

© Copyright Microsoft Corporation. All rights reserved.