Put It in Writing

Jeff Hicks
AUTHOR ~ TEACHER ~ SENSEI

@jeffhicks https://jdhitsolutions.com/blog

Why Script?

Automate repetitive and low-value tasks

Ensure consistency

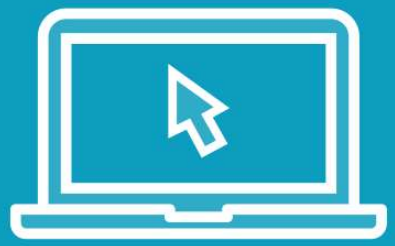
Manage at scale

Enhance your career

```
#SystemReport.ps1

<#
get a breakdown of error sources in the
System event log
#>

$data = Get-Eventlog System -EntryType
Error -Newest 1000 |
Group Source -NoElement

$title = "System Log Analysis"
$css = "./sample.css"

$data | Sort Count,Name -Desc |
Select Count,Name |
ConvertTo-Html -Title $Title -Pre
"<H1>$Computername</H1>" -CssUri $css |
Out-File c:\work\systemsources.html
```

- A basic script is a text file with a .ps1 file extension
- Use # or <#..#> to indicate comments
- Runs from start to finish
- Runs with your credentials

Demo



Creating a basic script

Challenges

Script is static

Hard-coded values

Requires editing when a change is needed

Parameterize

Avoid hard-coding values

Provide flexibility to the user

You can set defaults

```
Param (
[string]$LogName,
[string]$Computername = $env:computername,
[string]$Path,
[int32]$Newest = 500
)
```

Recommended to specify an object type

Adding Parameters

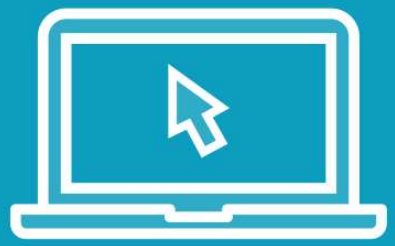
Use Param () at beginning of your script

Define a variable for each parameter

- Separated by a comma

You can set a default value

Demo



Creating a parameterized script

Best Practices

Parameterize to avoid hard-coding

No hard coded passwords or credentials

Don’t use aliases

Use full cmdlet and parameter names

Format your code

Write for the next person...

- It could be you!

Summary

There is no difference between a script and commands entered at the console

A script saves time

A script means consistency

Automation can mean freedom