

DEV208: WINDOWS POWERSHELL FOR SHAREPOINT 2010 DEVELOPERS

Gary Lapointe, MVP

About Me



- SharePoint MVP
- Independent Consultant and Owner of Falchion Consulting, LLC
 - <http://www.falchionconsulting.com>
- Principal Consultant - Aptillon, Inc.
 - <http://www.aptillon.com>
- Blog: <http://blog.falchionconsulting.com/>
- Twitter: @glapointe
- Email: gary@falchionconsulting.com



Agenda

- Windows PowerShell 101
- Common Tasks
- Extending PowerShell
- Questions

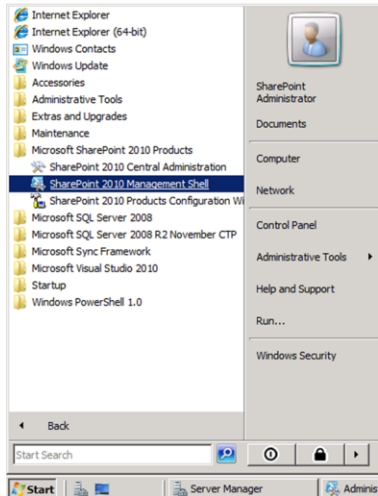
WINDOWS POWERSHELL 101

Getting on the same page...



BEST PRACTICES CONFERENCE SHAREPOINT ►

Management Shell



• `C:\Windows\System32\WindowsPowerShell\v1.0\PowerShell.exe -NoExit " & 'C:\Program Files\Common Files\Microsoft Shared\Web Server Extensions\14\CONFIG\POWER SHELL\Registration\sharepoint.ps1' "`

Load Snap-in For any Editor

```
1 if (!(Test-Path $profile.AllUsersAllHosts)) {
2   New-Item -Type file -Path $profile.AllUsersAllHosts -Force
3   $sb = @"
4   if (`$host.UI.RawUI.WindowTitle -ne "Administrator: SharePoint 2010 Management Shell") {
5     `$ver = `$host | select version
6     if (`$ver.Version.Major -gt 1) {$host.Runspace.ThreadOptions = "ReuseThread"}
7     if ((Get-PSSnapin "Microsoft.SharePoint.PowerShell" -ErrorAction SilentlyContinue) -eq `$null) {
8       Add-PSSnapin "Microsoft.SharePoint.PowerShell"
9     }
10  }
11  "@
12  Set-Content -Path $profile.AllUsersAllHosts -Value $sb
13 }
```



```
if (!(Test-Path $profile.AllUsersAllHosts)) {
  New-Item -Type file -Path $profile.AllUsersAllHosts -Force
}
$sb = @"
if (`$host.UI.RawUI.WindowTitle -ne "Administrator: SharePoint 2010 Management
Shell") {
  `$ver = `$host | select version
  if (`$ver.Version.Major -gt 1) {$host.Runspace.ThreadOptions = "ReuseThread"}
  if ((Get-PSSnapin "Microsoft.SharePoint.PowerShell" -ErrorAction SilentlyContinue) -
eq `$null) {
    Add-PSSnapin "Microsoft.SharePoint.PowerShell"
  }
}
"@
Set-Content -Path $profile.AllUsersAllHosts -Value $sb
```

Required Permissions

- Member of WSS_ADMIN_WPG and SharePoint_Shell_Access
- Use Add-SPShellAdmin to add a user to these groups

```
1 Add-SPShellAdmin -UserName domain\user
2
3 Get-SPDatabase | where {
4     $_.Name -eq "SharePoint_Content_*"
5 } | Add-SPShellAdmin -UserName domain\user
```

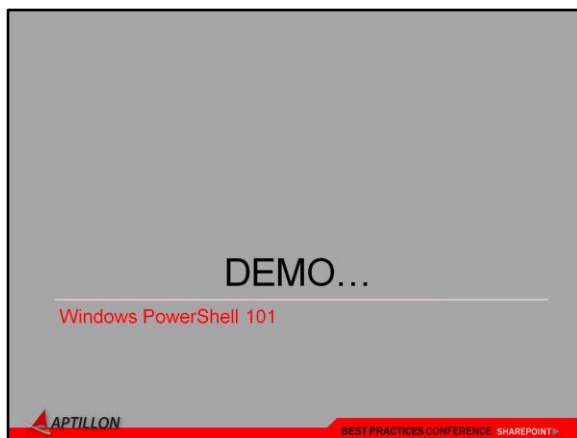
- Some commands require the user to be a local server admin and a Farm admin
- Remove-SPShellAdmin does not remove from WSS_ADMIN_WPG!



SharePoint_Shell_Access role exists in the config db by default but no other databases. The Add-SPShellAdmin cmdlet will add the user to the WSS_ADMIN_WPG group on each server (not just the executing server). The Remove-SPShellAdmin cmdlet does not remove from the WSS_ADMIN_WPG group so you must manually remove the user from this group.

To add to the Farm Admin group:

```
New-SPUser -UserAlias "domain\user" -Web "http://sp2010:1234/" -Group "Farm Administrators"
```



- Constructs (if/else/elseif, foreach, do/while, while, foreach-object)

```
function Test-Function ($val = "default val") { }
function Test-Function ($val = $(throw "required")) { }
```

Help about_comment_based_help

Help about_functions_advanced_parameters

```
function New-SPFarm () {
<#
  .Synopsis
    Creates a new SharePoint 2010 Farm
  .Description
    Creates a new SharePoint 2010 Farm using the provided XML settings file
  .Example
    New-SPFarm -ConfigFile c:\config.xml
  .Parameter ConfigFile
    The path to an XML file containing Farm configuration settings.
  .Link
    Get-SPFarm
#>
[CmdletBinding(DefaultParameterSetName="File")]
param (
  [Parameter(Mandatory=$true, Position=0, ParameterSetName="File")] [ValidateNotNullOrEmpty()]
  [string]$ConfigFile,

  [Parameter(Mandatory=$true, Position=0, ParameterSetName="Xml")] [ValidateNotNullOrEmpty()]
  [xml]$ConfigXml,

  [Parameter(Mandatory=$false)]
  [switch]$ConnectToExisting
)
[xml]$config = $null
switch ($PsCmdlet.ParameterSetName) {
  "File" { $config = Get-Content $ConfigFile; break }
  "Xml" { $config = $ConfigXml; break }
}
1..10 | % {sleep 2; Write-Host "Doing something clever..."}
```


}

COMMON TASKS

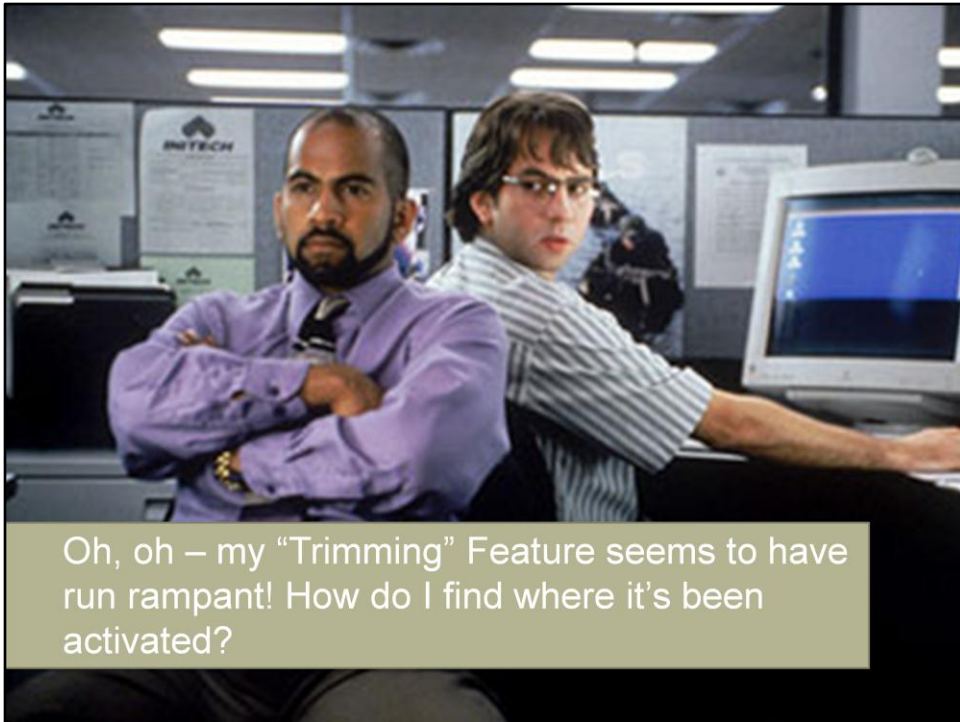
Getting things done...



BEST PRACTICES CONFERENCE SHAREPOINT ▶

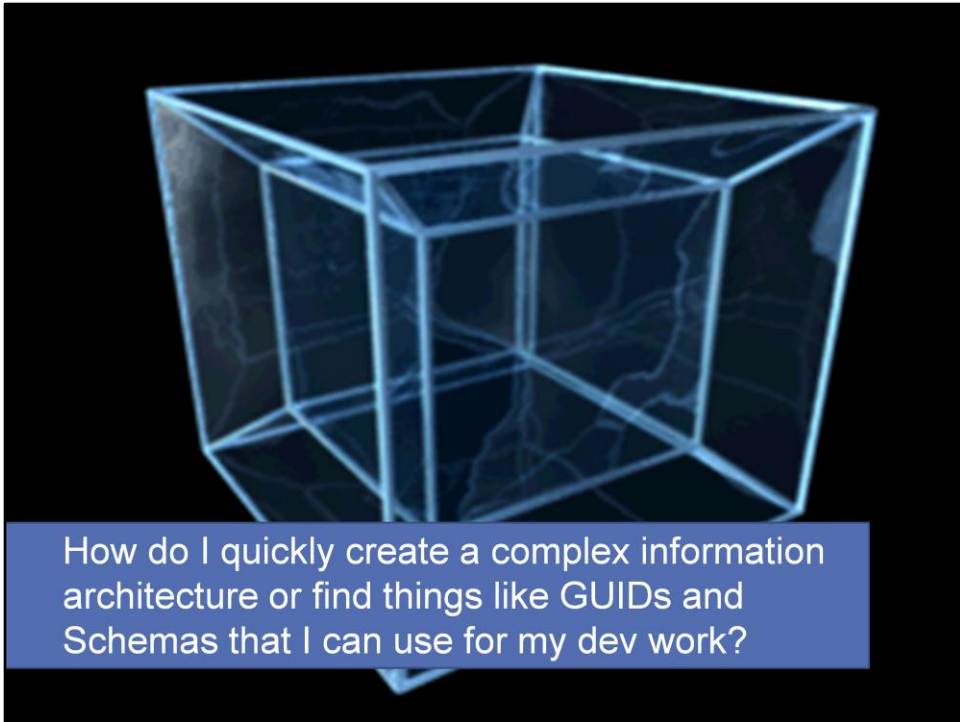


```
.\Deploy-SPSolutions.ps1  
Install-Solutions .\Solutions.xml
```



Oh, oh – my “Trimming” Feature seems to have run rampant! How do I find where it’s been activated?

Get-SPFeatureActivations.ps1
Get-SPFeatureActivations2.ps1



How do I quickly create a complex information architecture or find things like GUIDs and Schemas that I can use for my dev work?

```
$web.Fields | ? {$_.Group -eq "Custom Columns"} | %{$_.SchemaXml} | Out-File c:\fields.xml
```

```
$web.ContentTypes | ? {$_.Group -eq "Publishing Content Types"} | %{$_.SchemaXml} | Out-File c:\ct.xml
```



```
$dds =  
[Microsoft.SharePoint.Administration.SPWebService]::ContentService.DeveloperDash  
boardSettings  
$dds.DisplayLevel = "On"  
$dds.RequiredPermissions = "ManageWeb"  
$dds.Update()
```

EXTENDING POWERSHELL

Creating cmdlets, PipeBind objects, formatters, types,
and help files...



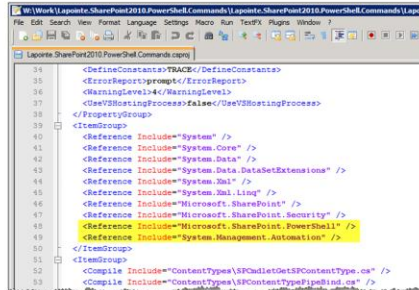
BEST PRACTICES CONFERENCE SHAREPOINT ►

Key Components

- Required XML registration file defines the custom cmdlets and maps the name to the class and help file
 - 14\Config\PowerShell\Registration\
- Optional XML help file
 - 14\Config\PowerShell\Help\
- Optional XML format file for defining views
 - 14\Config\PowerShell\Format\
- Optional XML types file allows declarative type extensions
 - 14\Config\PowerShell\Types\

Add Required Assemblies

- Add the following assembly references:
 - Microsoft.SharePoint.PowerShell
 - System.Management.Automation
- VS2010 does not show these assemblies in the references dialog so you must either browse to them or manually edit the project file



```
34 <DefineConstants>TRACE</DefineConstants>
35 <ErrorReport>prompt</ErrorReport>
36 <ManagedLevel>4</ManagedLevel>
37 <UseVSHostingProcess>False</UseVSHostingProcess>
38 </PropertyGroup>
39 </ItemGroup>
40 <Reference Include="System" />
41 <Reference Include="System.Core" />
42 <Reference Include="System.Data" />
43 <Reference Include="System.Data.DataSetExtensions" />
44 <Reference Include="System.Xml" />
45 <Reference Include="System.Xml.Linq" />
46 <Reference Include="Microsoft.SharePoint" />
47 <Reference Include="Microsoft.SharePoint.Security" />
48 <Reference Include="Microsoft.SharePoint.PowerShell" />
49 <Reference Include="System.Management.Automation" />
50 </ItemGroup>
51 </ItemGroup>
52 <Compile Include="ContentTypes\SPContentDefSPContentType.cs" />
53 <Compile Include="ContentTypes\SPContentTypePipeBind.cs" />
```

Base Classes

- SPCmdlet
- SPRemoveCmdletBase<TCmdletObject>
- SPSetCmdletBase<TCmdletObject>
- SPGetCmdletBase<TCmdletObject>
- SPNewCmdletBase<TCmdletObject>

- SPCmdletPipeBind<TCmdletObject>



DEMO...

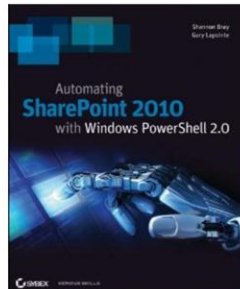
Extending PowerShell



BEST PRACTICES CONFERENCE SHAREPOINT ▶

Key Takeaways

- Learn PowerShell!!!
- Script your deployments
- Use PowerShell to understand your Feature footprint before upgrading
- Cmdlet interfaces are quicker to create than browser interfaces
- Buy my book!!!



About Aptillon



- SharePoint MVPs
- Microsoft Certified Master
- Consultants, Trainers, Authors, Speakers, Bloggers
- Great People, Great Skill, Great Talent, Great Passion



Todd Baginski



David Mann



Gary Lapointe



Darrin Bishop



Maurice Prather



Dan Holme



Matthew McDermott



QUESTIONS?

Thank You!!!