HDEV16: SharePoint 2010 PowerShell for Developers

Gary Lapointe, MVP
Falchion Consulting, LLC
gary@falchionconsulting.com

SharePoint CONNECTIONS

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://stsadm.blogspot.com/
- Twitter: @glapointe
- Email: gary@falchionconsulting.com

SharePoint CONNECTIONS

Agenda

- SharePoint 2010 PowerShell Overview
- Developer Cmdlets
- Using PowerShell with Visual Studio
- Developing Custom Cmdlets

SharePoint CONNECTIONS

Review SharePoint Specific PowerShell Concepts

POWERSHELL & SHAREPOINT

SharePoint CONNECTIONS

STSADM/PSConfig???

- They’re still there but the use of them has been, for the most part, made obsolete
- Most everything you can do with them can be done with PowerShell (and faster)
- You can access them via the PowerShell console (but why would you?)
- Extensions to STSADM are still supported (just recompile and fix any necessary bugs related to API changes)

SharePoint CONNECTIONS

Popular Editors

- SharePoint 2010 Management Shell
  - PowerShell Console
- Windows PowerShell ISE
  - Debugging
  - Tab Completion
  - Syntax Highlighting
- PowerGUI
  - Debugging
  - Tab Completion
  - Syntax Highlighting
  - Intellisense
- Visual Studio
  - Not so much – may as well just use notepad

SharePoint CONNECTIONS

Load Snap-in For any Editor

- Loading the SharePoint snap-in via your profile script allows you to use any editor
- Run the following to create a profile script if one doesn't exist and edit it in the ISE:

```
if (!(test-path $profile.AllUsersAllHosts)) {
    new-item -type file -path $profile.AllUsersAllHosts -force}

powershell_ise $profile.AllUsersAllHosts
```

- Add the following code to the script file and save your changes:

```
$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"
}
```

SharePoint CONNECTIONS

Required Permissions

- Member of WSS_ADMIN_WPG and SharePoint_Shell_Access
  - WSS_ADMIN_WPG is a local security group on the machine the user is executing commands on
  - SharePoint_Shell_Access is a SQL Role in the target Database(s)
- Use Add-SPShellAdmin to add a user to these groups

```
Add-SPShellAdmin -UserName domain\user

Get-SPDatabase | where {$_.Name -like
   "SharePoint_Content_*"} | Add-SPShellAdmin -UserName
   domain\user
```

- Some commands (such as setup commands) require the user to be a local server admin but most do not.

SharePoint CONNECTIONS

PipeBind Objects

- Allows different representations of an artifact to be passed into cmdlets
  - Example: SPSitePipeBind accepts either a GUID, URL, or SPSite object

```
Get-SPWeb [-Identity <SPWebPipeBind>] [-
   AssignmentCollection <SPAssignmentCollection>] [-
   Confirm [<SwitchParameter>]] [-Filter <ScriptBlock>] [-
   Limit <String>] [-Regex <SwitchParameter>] [-Site
   <SPSitePipeBind>] [-WhatIf [<SwitchParameter>]]
   [<CommonParameters>]

$site = Get-SPSite "http://portal"
$webs = $site | Get-SPWeb
$webs = $site.ID | Get-SPWeb
$webs = "http://portal" | Get-SPWeb
```

Handling Disposable Items

- SPAssignmentCollection
  - Collection object that all SharePoint cmdlets can accept to store objects that must be disposed
    - Get-SPSite [-AssignmentCollection <SPAssignmentCollection>] [-Confirm [<SwitchParameter>]] [-Filter <ScriptBlock>] [-Limit <String>] [-WebApplication <SPWebApplicationPipeBind>] [-WhatIf [<SwitchParameter>]] [<CommonParameters>]

SharePoint CONNECTIONS

Assignment Collections

- Start-SPAssignment
  - Start-SPAssignment [-AssignmentCollection <SPAssignmentCollection>] [-Global <SwitchParameter>] [<CommonParameters>]
- Stop-SPAssignment
  - Stop-SPAssignment [-SemiGlobal <SPAssignmentCollection>] [-AssignmentCollection <SPAssignmentCollection>] [-Global <SwitchParameter>] [<CommonParameters>]
- Three levels of assignment:
  - No assignment (dispose immediately)
  - Simple assignment (use a global store)
  - Advanced assignment (use a named store)
- Only Get-SPSite, Get-SPWeb, New-SPSite, and New-SPWeb use this disposal capability

SharePoint CONNECTIONS

Assignment Collection Example

- *-Global* stores items within an internal static variable
- *-SemiGlobal* is used for named variables and can be passed via the pipeline

```
#Use of the Global assignment variable
Start-SPAssignment -Global
$site = Get-SPSite "http://portal"
$site | fl
Stop-SPAssignment -Global

#Use of a semi-global, or named variable
$gc = Start-SPAssignment
$site = $gc | Get-SPSite "http://mysites"
$site | fl
$gc | Stop-SPAssignment
```

SharePoint CONNECTIONS

Using SharePoint Cmdlets

DEMO

SharePoint CONNECTIONS

Developer Dashboard and Solution Deployment

DEVELOPER CMDLETS

SharePoint CONNECTIONS

Developer Dashboard

- The developer dashboard enables you to view performance and debugging data for a given page request.

```
$dash =
    [Microsoft.SharePoint.Administration.SPWebService]::
    ContentService.DeveloperDashboardSettings
$dash.DisplayLevel = "OnDemand"
$dash.TraceEnabled = $true
$dash.Update()
```

SharePoint CONNECTIONS

Solution Deployment Cmdlets

- Get-SPSolution
- Add-SPSolution
  - stsadm -o addsolution
- Install-SPSolution
  - stsadm -o deploysolution
- Uninstall-SPSolution
  - stsadm -o retractsolution
- Remove-SPSolution
  - stsadm -o deletesolution
- Update-SPSolution
  - stsadm -o upgradesolution

*There are also existing User Solution cmdlets

- Get-SPFeature
- Enable-SPFeature
  - stsadm -o activatefeature
- Disable-SPFeature
  - stsadm -o deactivatefeature
- Install-SPFeature
  - stsadm -o installfeature
- Uninstall-SPFeature
  - stsadm -o uninstallfeature

SharePoint CONNECTIONS

Solution Deployment

DEMO

SharePoint CONNECTIONS

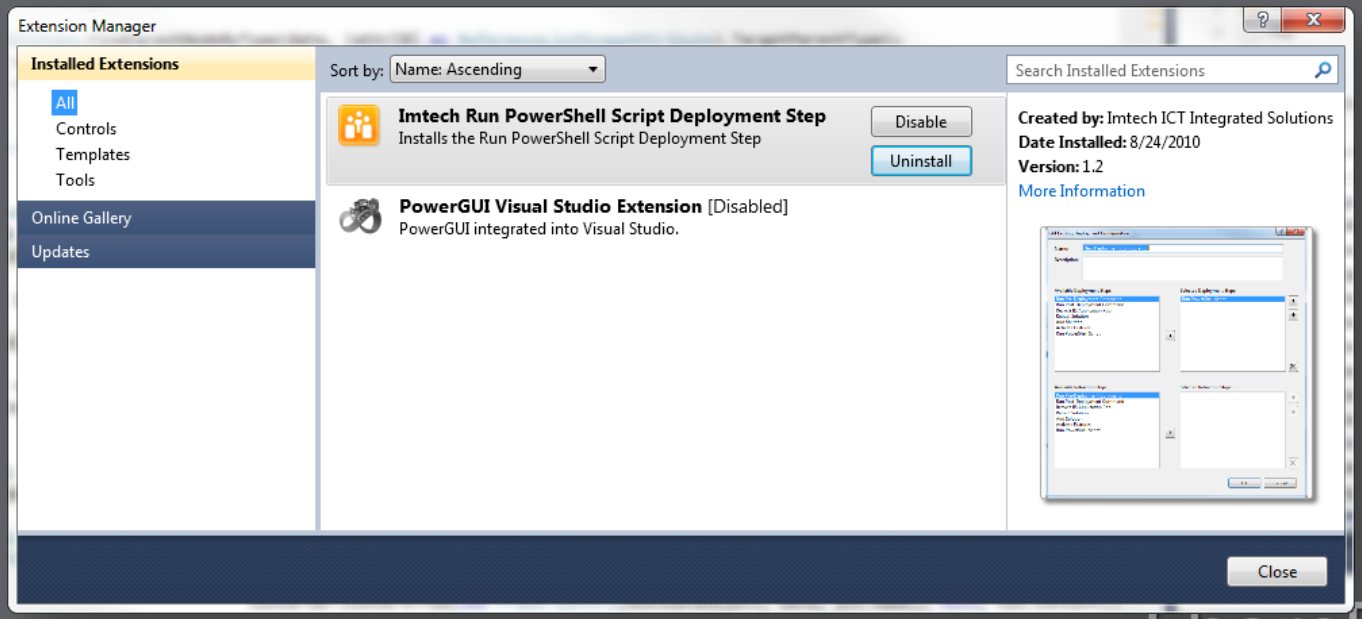
Executing Scripts at Build

USING POWERSHELL WITH VISUAL STUDIO 2010

SharePoint CONNECTIONS

PowerShell Script Deployment Step

- Developed by Waldek Mastykarz
  - Integrated into CKS:DEV
  - http://cksdev.codeplex.com/



SharePoint CONNECTIONS

Pre/Post Build Events

- May need to call the SysNative version of PowerShell (x86 vs. x64 issues)
  - C:\windows\sysnative\WindowsPowerShell\v1.0\powershell.exe
- Example from my cmdlets
  - powershell.exe -command [System.Reflection.Assembly]::LoadFile('$(TargetPath)');[$(ProjectName).MamlGenerator.CmdletHelpGenerator]::GenerateHelp('$(ProjectDir)POWERSHELL\Help', $true)

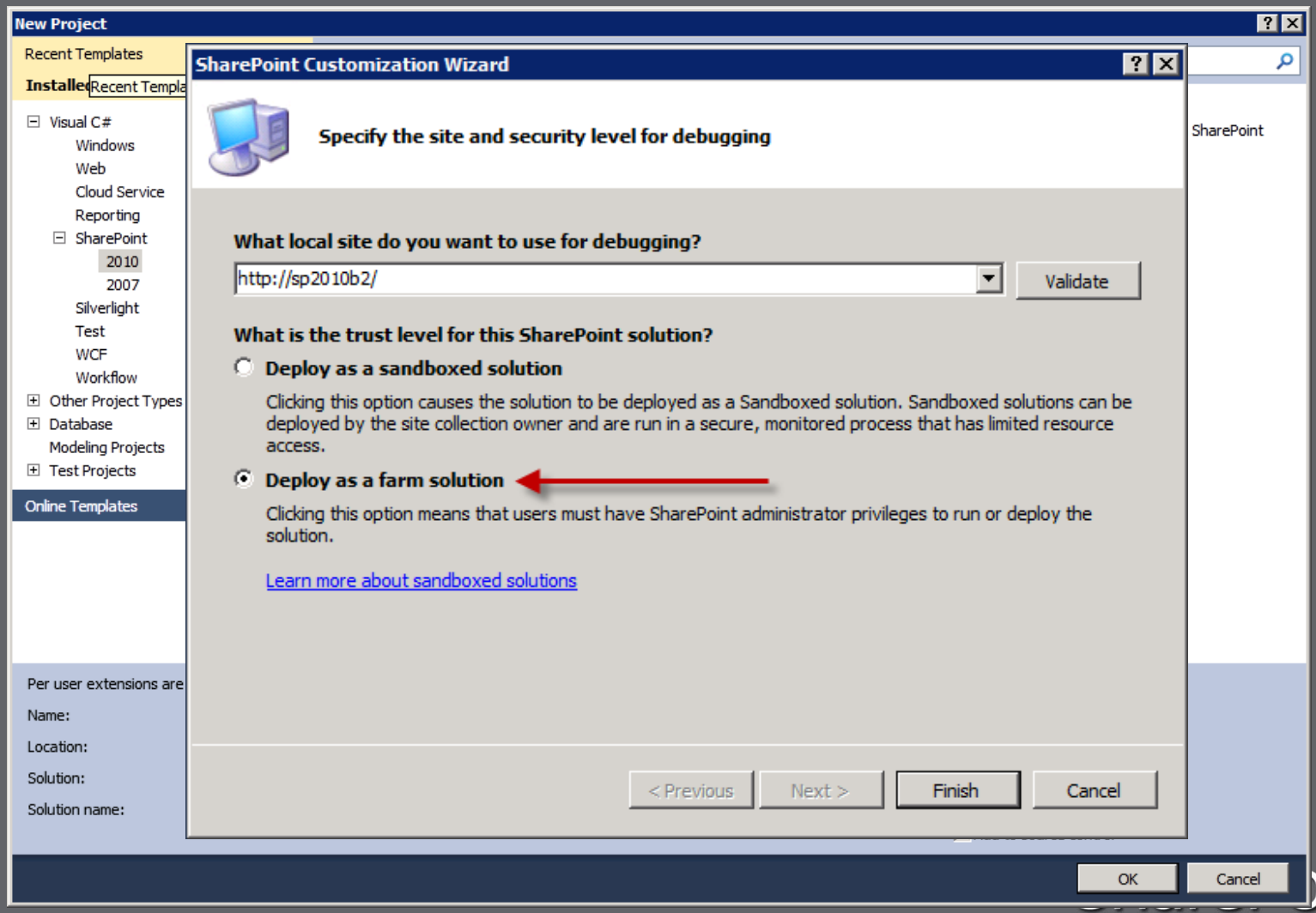
SharePoint CONNECTIONS

Visual Studio 2010 Cmdlet Development

BUILDING CUSTOM CMDLETS

SharePoint CONNECTIONS

Creating a new VS2010 Project



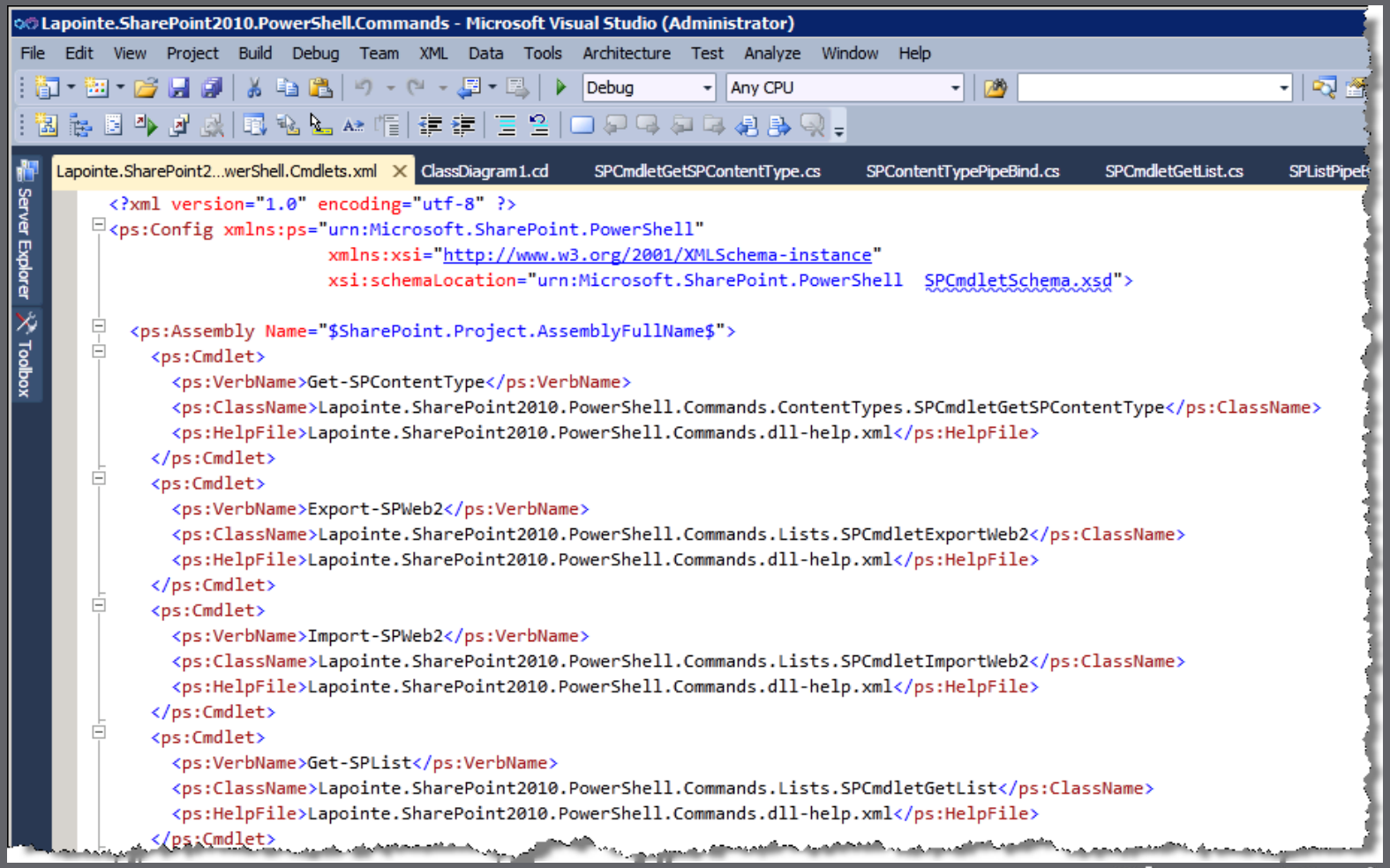
CONNECTIONS

Key Components

- Required XML registration file defines the custom cmdlets and maps the name to the class and help file
  - 14\Config\PowerShell\Registration\<project name>.xml
- Optional XML help file
  - 14\Config\PowerShell\Help\<dll name>-help.xml
- Optional XML format file
  - 14\Config\PowerShell\Format\<project name>.Format.ps1xml
- Optional XML types file allows aliased properties to be declaratively added to existing types
  - 14\Config\PowerShell\Types\<project name>.ps1xml

SharePoint CONNECTIONS

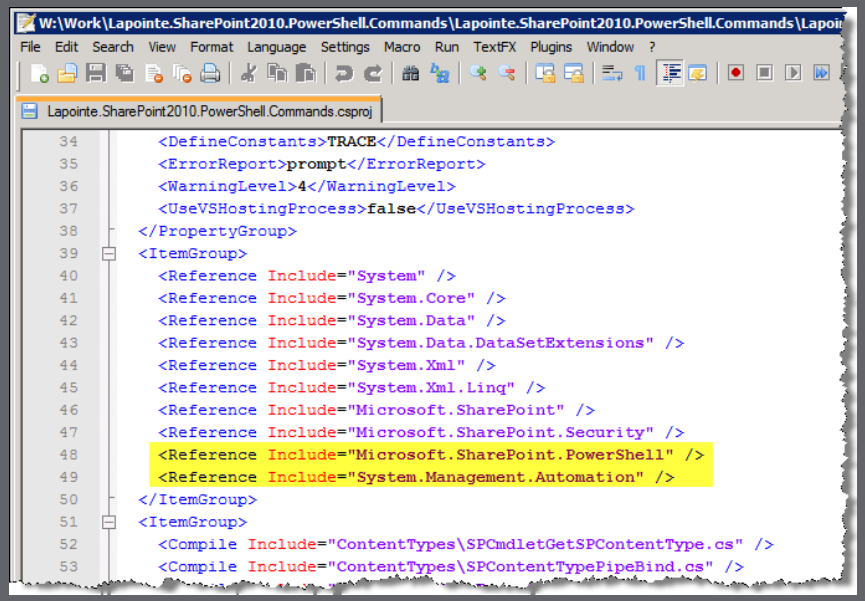
Registration XML



SharePoint CONNECTIONS

Add Required Assemblies

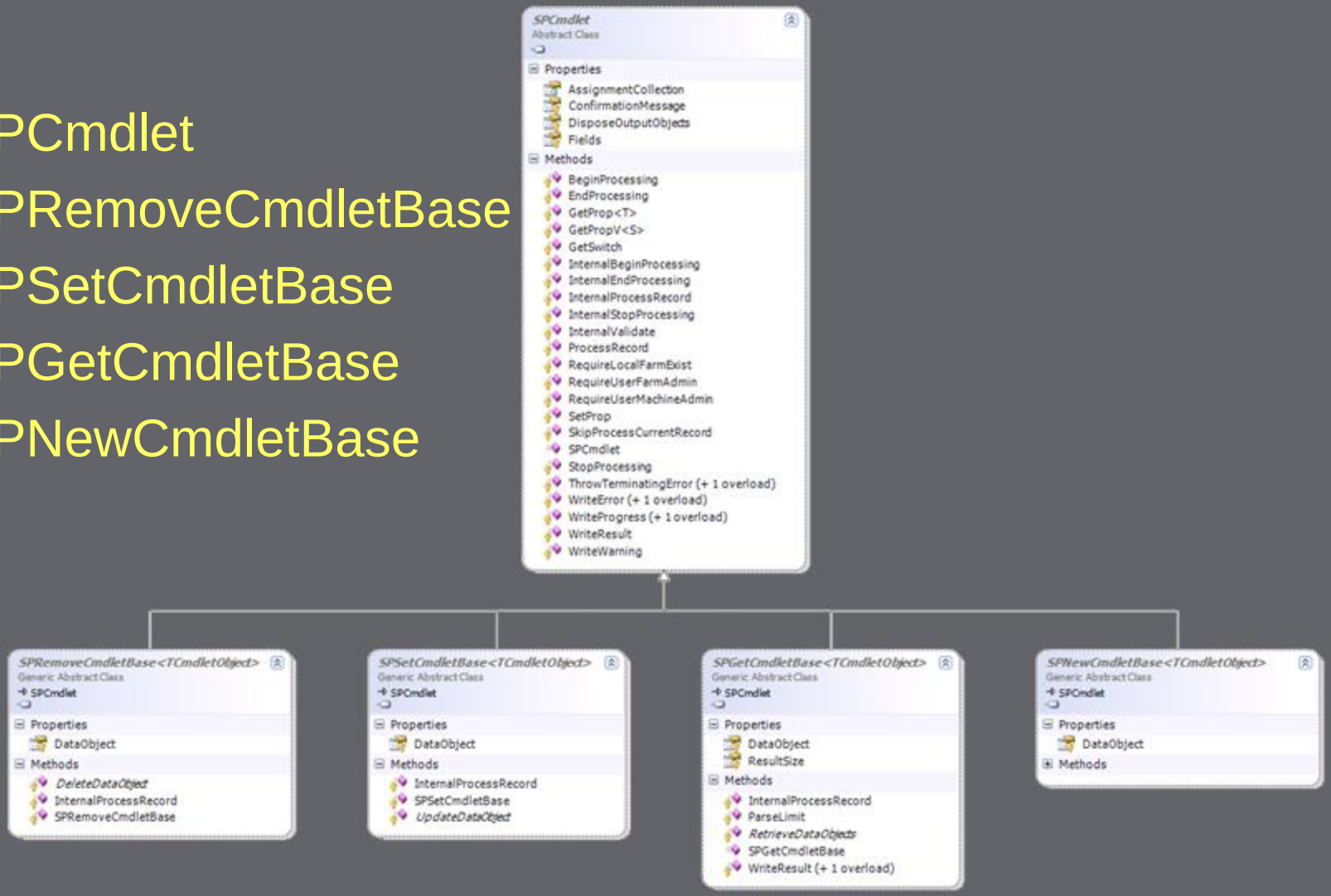
- Add the following assembly references:
  - Microsoft.SharePoint.PowerShell
  - System.Management.Automation
- VS2010 does not show these assemblies in the references dialog so you must either browse to them or manually edit the project file



SharePoint CONNECTIONS

Cmdlet Base Classes

- SPCmdlet
- SPRemoveCmdletBase
- SPSetCmdletBase
- SPGetCmdletBase
- SPNewCmdletBase



CONNECTIONS

Creating Custom Cmdlets

DEMO

SharePoint CONNECTIONS

Your Feedback is Important

Please fill out a session evaluation form drop it off at the conference registration desk.

Thank you!

SharePoint CONNECTIONS