POWERSHELL WORKFLOW BASICS

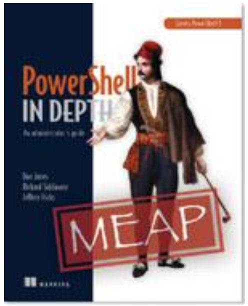
Jeffery Hicks

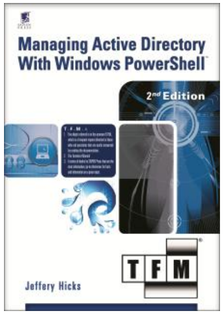
Windows PowerShell MVP

jhicks@jdhitsolutions.com

WHO AM I?

- Windows PowerShell MVP
- PowerShell Author
  - PowerShell in Depth (with Don Jones and Richard Siddaway)
  - Windows PowerShell 2.0: TFM (with Don Jones)
  - Managing Active Directory with Windows PowerShell: TFM 2nd Ed.
- IT trainer and consultant
- http://jdhitsolutions.com/blog
- http://twitter.com/jeffhicks





AGENDA

- Definitions
- Requirements
- Limitations
- Building Workflows
- Syntax
- Resources
- Q&A

A NOTE…

- All demos will be made available
- I'm using a beta of PowerShell 3.0 so no guarantees

DEFINITIONS - WHAT IS A WORKFLOW?

- A robust multi-machine orchestration engine
- Designed for long running unattended tasks across potentially thousands of machines
- Persistent states can survive reboots and network interruptions
- Can be integrated with WCF Services and AppFabric

DEFINITIONS - WHAT IS A WORKFLOW?

- Robust
  - Persistence via check points
  - Suspend and Resume capabilities
- Performance and Scalability
  - Parallel tasks
  - Connection pooling
  - Connection throttling
- PowerShell Based
  - Use existing cmdlets
  - No need to master XAML
  - Built in parameters for multi-machine management

DEFINITIONS - WHAT IS A WORKFLOW?

- Workflow activities are isolated
- All data and objects are serialized
- Objects are strongly typed
- Static scoping

WORKFLOW SCENARIOS

- Server deployment
- Server configuration/remediation
- User provisioning
- Private cloud deployments
- Sharepoint configuration
- Any business workflow that can be orchestrated with command line tools.

REQUIREMENTS

- Built on .NET Framework 4.0 and Windows Workflow Foundation
- Requires PowerShell 3.0
- Requires PowerShell 3.0 remoting
- Leverages PowerShell's job infrastructure

REQUIREMENTS: REMOTING

- Workflows connect to machines using WSMan protocol
- Connects to the default Workflow session configuration

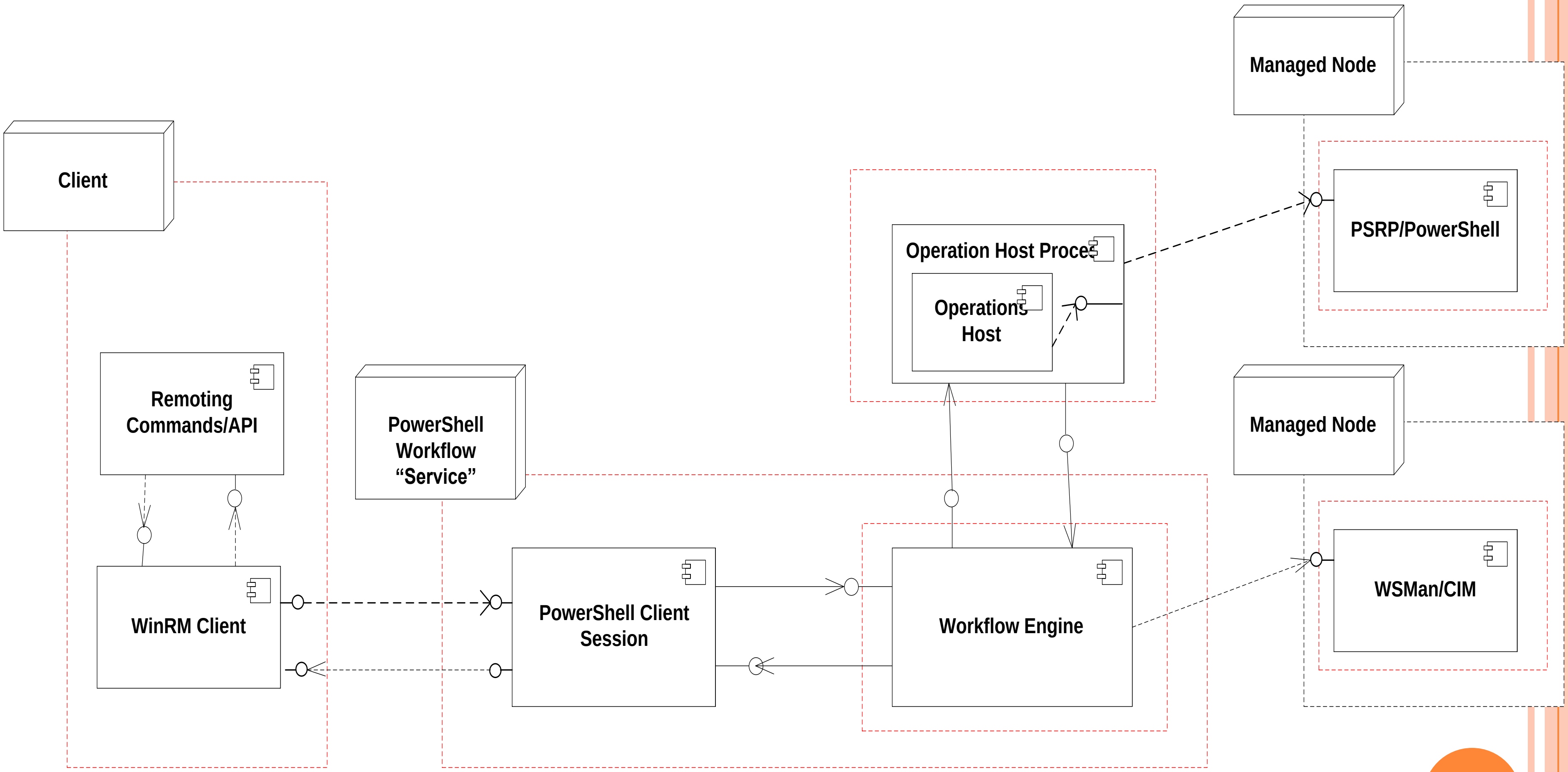
```
PS C:\> get-pssessionconfiguration
microsoft.powerShell.workflow
```

- Important defaults:
  - Max persistence storage 10GB
  - Max memory per shell 1GB
  - Admin permissions required
- Be very careful of changing this configuration

LIMITATIONS AND GOTCHAS

- All objects and data must be "serializable"
- Must use full cmdlet ***and*** parameter names
- No positional parameters
- No Begin/Process/End scriptblocks
- No "eventing"
- No Traps - Use Try/Catch
- Not intended to be interactive - can't use Write-Host.
- On comment based help - must use maml
- Pay close attention to scope!

WORKFLOW ARCHITECTURE



BUILDING WORKFLOWS

- Don't replace "Function" with "Workflow"
- Start new and plan out your activities
- Minimize sharing of data or variables across activities
- PowerShell turns your workflow into XAML
- Workflow is a command type

```
PS C:\> get-command -commandtype Workflow
```

SYNTAX: INLINESCRIPT

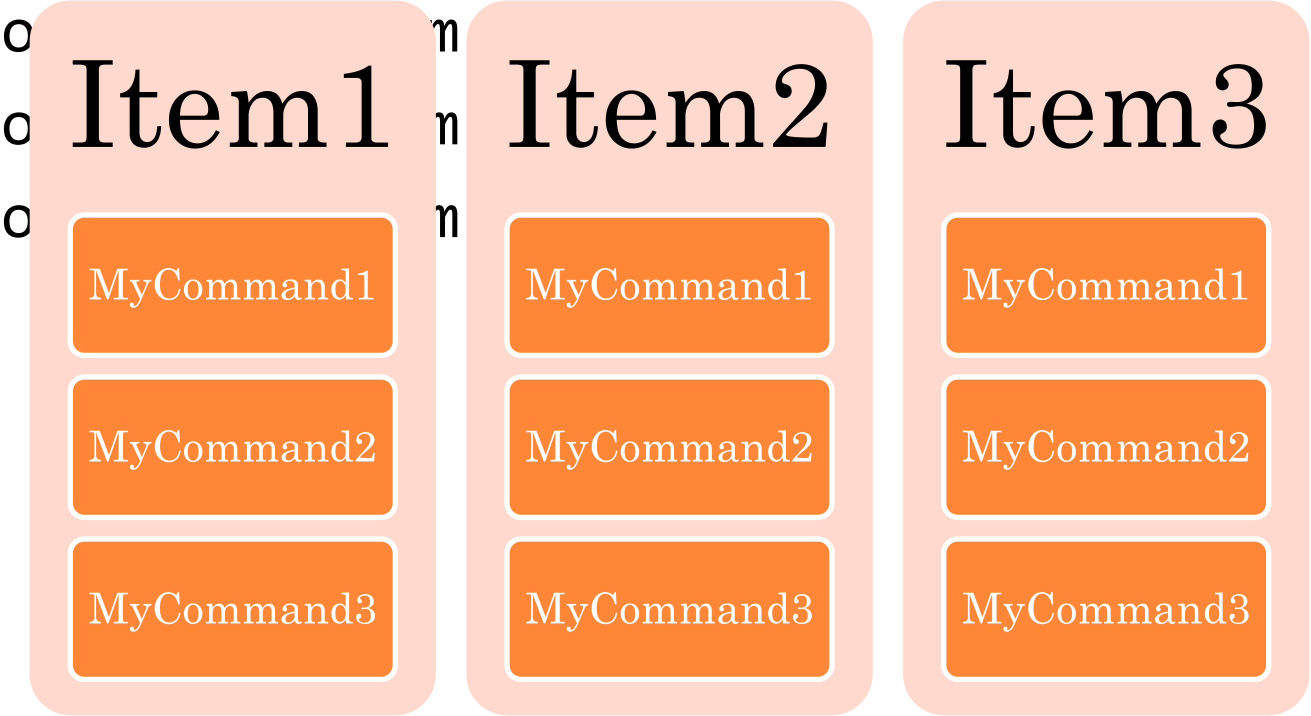
- Send PowerShell commands to remote machine(s)
- Runs out-of-process
- Runtime command validation at runtime
- This is really a series of Invoke-Command activities

SYNTAX: PARALLEL

- Execute a collection of activities independently and in parallel
- Foreach -Parallel
  - The parameter only works in a workflow
  - Run a set of commands in parallel for each object in a collection
- Parallel key word
  - Run a set of commands simultaneously and in parallel
  - Runs in a new scope
  - Often used to run a series of Sequences

FOREACH -PARALLEL

```
Foreach -parallel ($item in $objects) {
   MyCo              m
   MyCo              m
   MyCo              m
}
```



PARALLEL

```
Parallel {
        MyCommand1
        MyCommand
        MyCommand
}
```

MyCommand1

MyCommand2

MyCommand3

SYNTAX: SEQUENCE

- Execute a collection of tasks in order
- Often used with Parallel
- Watch out for scope!

SYNTAX: SEQUENCE

```
Parallel {
$var=123
     Sequence {
          MyCommand1 $workflow:var
     }
     Sequence {
          MyCommand2 $workflow:var
     }
}
```

SYNTAX: SCOPE AND VARIABLES

- Workflows uses static scopes, i.e. PowerShell won't "search" for a variable
- Can use $Workflow:myvar
- Access "out of scope" variables with $Using:MyVar
  - Read-only
  - Available from InlineScript

SYNTAX: ASJOB

- Run any workflow with -asjob
- Use job cmdlets to manage
- Import PSWorkFlow module to add new job type definitions
- Useful with suspended workflows

SYNTAX: COMMON PARAMETERS

- All workflows have a set of common parameters
- They do not need to be defined in the workflow
  - PSComputerName
  - PSCredential
  - PSConnectionRetryCount
  - PSActionRetryCount
  - PSPersist

SYNTAX: COMMON DATA

- Result
- PSUserName
- PSVerbose
- WorkflowCommandName
- PSProgress
- PSWarning
- PSError
- PSDebug
- JobName
- PSWorkflowPath
- Input

SYNTAX: PERSISTENCE

- Workflows can be made persistent to survive interruptions
- By default restarts the workflow unless…
- Set persistence per activity
  - Checkpoint-Workflow
  - Persist
  - Suspend-Workflow
- Set persistence for the entire workflow
  - -PSPersist common parameter
  - Set to $True: -pspersist $true

DEMOS

MOVING ON

- Nested workflows
- Workflows calling workflows
- Suspending and resuming
- Troubleshooting and debugging
- Importing workflows from Visual Studio Workflow Designer

Questions

MORE RESOURCES

- PowerShell in Depth: An Administrators Guide by Don Jones, Richard Siddaway and Jeffery Hicks (Manning Press, in production)
- Learn PowerShell in a Month of Lunches, 2nd Ed. By Don Jones and Jeffery Hicks (Manning Press, in production)
- Windows PowerShell Team blog (http://blogs.msdn.com/powershell)
- The Lonely Administrator (http://jdhitsolutions.com/blog)
- Prof. PowerShell (http://mcpmag.com/articles/list/prof-powershell.aspx

THANK YOU

- http://jdhitsolutions.com/blog
- jhicks@jdhitsolutions.com