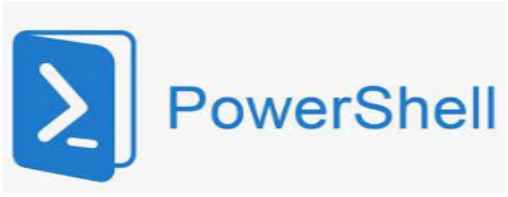

Full credit is given to the above companies including the OS that this PDF file was generated!

PowerShell Get-Help on command 'Register-PSSessionConfiguration'

PS C:\Users\wahid> Get-Help Register-PSSessionConfiguration

NAME

Register-PSSessionConfiguration

SYNOPSIS

Creates and registers a new session configuration.

SYNTAX

Register-PSSessionConfiguration [-Name] <System.String> [-AccessMode {Disabled | Local | Remote}] [-ApplicationBase <System.String>] [-Force] [-MaximumReceivedDataSizePerCommandMB <System.Nullable`1[System.Double]>] [-MaximumReceivedObjectSizeMB <System.Nullable`1[System.Double]>] [-ModulesToImport <System.Object[]>] [-NoServiceRestart] [-ProcessorArchitecture {x86 | amd64}] [-PSVersion <System.Version>] [-RunAsCredential <System.Management.Automation.PSCredential>] [-SecurityDescriptorSddl <System.String>] [-SessionType {DefaultRemoteShell | Workflow}] [-SessionTypeOption <System.Management.Automation.PSSessionTypeOption>] [-ShowSecurityDescriptorUI] [-StartupScript <System.String>] [-ThreadApartmentState {STA | MTA | Unknown}] [-ThreadOptions {Default | UseNewThread | ReuseThread | UseCurrentThread}] [-TransportOption

<System.Management.Automation.PSTransportOption>] [-UseSharedProcess]
[-Confirm] [-WhatIf] [<CommonParameters>]

Register-PSSessionConfiguration [-Name] <System.String> [-AssemblyName]
<System.String> [-ConfigurationTypeName] <System.String> [-AccessMode
{Disabled | Local | Remote}] [-ApplicationBase <System.String>] [-Force]
[-MaximumReceivedDataSizePerCommandMB <System.Nullable`1[System.Double]>]
[-MaximumReceivedObjectSizeMB <System.Nullable`1[System.Double]>]
[-ModulesToImport <System.Object[]>] [-NoServiceRestart]
[-ProcessorArchitecture {x86 | amd64}] [-PSVersion <System.Version>]
[-RunAsCredential <System.Management.Automation.PSCredential>]
[-SecurityDescriptorSddl <System.String>] [-SessionTypeOption
<System.Management.Automation.PSSessionTypeOption>]
[-ShowSecurityDescriptorUI] [-StartupScript <System.String>]
[-ThreadApartmentState {STA | MTA | Unknown}] [-ThreadOptions {Default |
UseNewThread | ReuseThread | UseCurrentThread}] [-TransportOption
<System.Management.Automation.PSTransportOption>] [-UseSharedProcess]
[-Confirm] [-WhatIf] [<CommonParameters>]

Register-PSSessionConfiguration [-Name] <System.String> [-AccessMode {Disabled
| Local | Remote}] [-Force] [-MaximumReceivedDataSizePerCommandMB
<System.Nullable`1[System.Double]>] [-MaximumReceivedObjectSizeMB
<System.Nullable`1[System.Double]>] [-NoServiceRestart] -Path <System.String>
[-ProcessorArchitecture {x86 | amd64}] [-RunAsCredential
<System.Management.Automation.PSCredential>] [-SecurityDescriptorSddl
<System.String>] [-ShowSecurityDescriptorUI] [-StartupScript <System.String>]
[-ThreadApartmentState {STA | MTA | Unknown}] [-ThreadOptions {Default |
UseNewThread | ReuseThread | UseCurrentThread}] [-TransportOption
<System.Management.Automation.PSTransportOption>] [-UseSharedProcess]
[-Confirm] [-WhatIf] [<CommonParameters>]

DESCRIPTION

The `Register-PSSessionConfiguration` cmdlet creates and registers a new session configuration on the local computer. This is an advanced cmdlet that you can use to create custom sessions for remote users.

Every PowerShell session (PSSession) uses a session configuration, also known as an endpoint. When users create a session that connects to the computer, they can select a session configuration or use the default session configuration that is registered when you enable PowerShell remoting. Users can also set the $PSSessionConfigurationName preference variable, which specifies a default configuration for remote sessions created in the current session.

The session configuration defines the environment for the remote session. The configuration can determine which commands and language elements are available in the session, and it can include settings that protect the computer, such as those that limit the amount of data that the session can receive remotely in a single object or command. The security descriptor of the session configuration determines which users have permission to use the session configuration.

You can define the elements of configuration by using an assembly that implements a new configuration class and by using a script that runs in the session. Beginning in PowerShell 3.0, you can also use a session configuration file to define the session configuration.

For information about session configurations, see about_Session_Configurations (About/about_Session_Configurations.md). For information about session configuration files, see about_Session_Configuration_Files (About/about_Session_Configuration_Files.md).

PARAMETERS

-AccessMode

<System.Management.Automation.Runspaces.PSSessionConfigurationAccessMode>

Enables and disables the session configuration and determines whether it can be used for remote or local sessions on the computer. The acceptable values for this parameter are:

- Disabled. Disables the session configuration. It cannot be used for remote or local access to the computer. - Local. Allows users of the local computer to use the session configuration to create a local loopback session on the same computer, but denies access to remote users. - Remote. Allows local and remote users to use the session configuration to create sessions and run commands on this computer.

The default value is Remote.

Other cmdlets can override the value of this parameter later. For example, the `Enable-PSRemoting` cmdlet allows for remote access to all session configurations, the `Enable-PSSessionConfiguration` cmdlet enables session configurations, and the `Disable-PSRemoting` cmdlet prevents remote access to all session configurations.

This parameter was introduced in PowerShell 3.0.

-ApplicationBase <System.String>

Specifies the path of the assembly file (*.dll) that is specified in the value of the AssemblyName parameter. Use this parameter when the value of the AssemblyName parameter does not include a path. The default is the current directory.

-AssemblyName <System.String>

Specifies the name of an assembly file (*.dll) in which the configuration type is defined. You can specify the path of the .dll in this parameter or in the value of the ApplicationBase parameter.

This parameter is required when you specify the ConfigurationTypeName

parameter.

-ConfigurationTypeName <System.String>
Specifies the fully qualified name of the Microsoft .NET Framework type
that is used for this configuration. The type that you specify must
implement the System.Management.Automation.Remoting.PSSessionConfiguration
class.

To specify the assembly file (*.dll) that implements the configuration
type, specify the AssemblyName and ApplicationBase parameters.

Creating a type lets you control more aspects of the session
configuration, such as exposing or hiding certain parameters of cmdlets,
or setting data size and object size limits that users cannot override.

If you omit this parameter, the DefaultRemotePowerShellConfiguration class
is used for the session configuration.

-Force <System.Management.Automation.SwitchParameter>
Suppresses all user prompts and restarts the WinRM service without
prompting. Restarting the service makes the configuration change effective.

To prevent a restart and suppress the restart prompt, specify the
NoServiceRestart parameter.

-MaximumReceivedDataSizePerCommandMB <System.Nullable`1[System.Double]>
Specifies a limit for the amount of data that can be sent to this computer
in any single remote command. Enter the data size in megabytes (MB). The
default is 50 MB.

If a data size limit is defined in the configuration type that is
specified in the ConfigurationTypeName parameter, the limit in the
configuration type is used and the value of this parameter is ignored.

-MaximumReceivedObjectSizeMB <System.Nullable`1[System.Double]>

Specifies a limit for the amount of data that can be sent to this computer in any single object. Enter the data size in megabytes. The default is 10 MB.

If an object size limit is defined in the configuration type that is specified in the ConfigurationTypeName parameter, the limit in the configuration type is used and the value of this parameter is ignored.

-ModulesToImport <System.Object[]>

Specifies the modules that are automatically imported into sessions that use the session configuration.

By default, only Microsoft.PowerShell.Core is imported into sessions. Unless the cmdlets are excluded, you can use `Import-Module` to add modules to the session.

The modules specified in this parameter value are imported in additions to modules that are specified by the SessionType parameter and those listed in the ModulesToImport key in the session configuration file (`New-PSSessionConfigurationFile`). However, settings in the session configuration file can hide the commands exported by modules or prevent users from using them.

This parameter was introduced in PowerShell 3.0.

-Name <System.String>

Specifies a name for the session configuration. This parameter is required.

-NoServiceRestart <System.Management.Automation.SwitchParameter>

Does not restart the WinRM service, and suppresses the prompt to restart the service.

By default, when you run a `Register-PSSessionConfiguration` command, you are prompted to restart the WinRM service to make the new session configuration effective. Until the WinRM service is restarted, the new session configuration is not effective.

To restart the WinRM service without prompting, specify the Force parameter. To restart the WinRM service manually, use the `Restart-Service` cmdlet.

-Path <System.String>

Specifies the path and filename of a session configuration file (.pssc), such as one created by `New-PSSessionConfigurationFile`. If you omit the path, the default is the current directory.

This parameter was introduced in PowerShell 3.0.

-ProcessorArchitecture <System.String>

Determines whether a 32-bit or 64-bit version of the PowerShell process is started in sessions that use this session configuration. The acceptable values for this parameter are: x86 (32-bit) and AMD64 (64-bit). The default value is determined by the processor architecture of the computer that hosts the session configuration.

You can use this parameter to create a 32-bit session on a 64-bit computer. Attempts to create a 64-bit process on a 32-bit computer fail.

-PSVersion <System.Version>

Specifies the version of PowerShell in sessions that use this session configuration.

The value of this parameter takes precedence over the value of the PowerShellVersion key in the session configuration file.

This parameter was introduced in PowerShell 3.0.

-RunAsCredential <System.Management.Automation.PSCredential>
Specifies credentials for commands in the session. By default, commands run with the permissions of the current user.

This parameter was introduced in PowerShell 3.0.

-SecurityDescriptorSddl <System.String>
Specifies a Security Descriptor Definition Language (SDDL) string for the configuration.

This string determines the permissions that are required to use the new session configuration. To use a session configuration in a session, users must have at least Execute (Invoke) permission for the configuration.

If the security descriptor is complex, consider using the ShowSecurityDescriptorUI parameter instead of this parameter. You cannot use both parameters in the same command.

If you omit this parameter, the root SDDL for the WinRM service is used for this configuration. To view or change the root SDDL, use the WSMan provider. For example `Get-Item wsman:\localhost\service\rootSDDL`. For more information about the WSMan provider, type `Get-Help wsman`.

-SessionType <System.Management.Automation.Runspaces.PSSessionType>
Specifies the type of session that is created by using the session configuration. The acceptable values for this parameter are:

- Empty. No modules are added to session by default. Use the parameters of this cmdlet to add modules, functions, scripts, and other features to the session. - Default. Adds Microsoft.PowerShell.Core to the session.

This module includes the `Import-Module` cmdlet that users can use to import other modules unless you explicitly prohibit the cmdlet. - RestrictedRemoteServer. Includes only the following cmdlets: `Exit-PSSession`, `Get-Command`, `Get-FormatData`, `Get-Help`, `Measure-Object`, `Out-Default`, and `Select-Object`. Use a script or assembly, or the keys in the session configuration file, to add modules, functions, scripts, and other features to the session.

The default value is Default.

The value of this parameter takes precedence over the value of the SessionType key in the session configuration file.

This parameter was introduced in PowerShell 3.0.

-SessionTypeOption <System.Management.Automation.PSSessionTypeOption>
Specifies type-specific options for the session configuration. Enter a session type options object, such as the PSWorkflowExecutionOption object that the `New-PSWorkflowExecutionOption` cmdlet returns.

The options of sessions that use the session configuration are determined by the values of session options and the session configuration options. Unless specified, options set in the session, such as by using the `New-PSSessionOption` cmdlet, take precedence over options set in the session configuration. However, session option values cannot exceed maximum values set in the session configuration.

This parameter was introduced in PowerShell 3.0.

-ShowSecurityDescriptorUI <System.Management.Automation.SwitchParameter>
Indicates that this cmdlet displays a property sheet that helps you create the SDDL for the session configuration. The property sheet appears after you enter the `Register-PSSessionConfiguration` command and then restart

the WinRM service.

When setting the permissions for the configuration, remember that users must have at least Execute (Invoke) permission to use the session configuration in a session.

You cannot use the SecurityDescriptorSDDL parameter and this parameter in the same command.

-StartupScript <System.String>
Specifies the fully qualified path of a PowerShell script. The specified script runs in the new session that uses the session configuration.

You can use the script to additionally configure the session. If the script generates an error, even a non-terminating error, the session is not created and the `New-PSSession` command fails.

-ThreadApartmentState <System.Threading.ApartmentState>
Specifies the apartment state of the threads in the session. The acceptable values for this parameter are: STA, MTA, and Unknown. The default value is Unknown.

-ThreadOptions <System.Management.Automation.Runspaces.PSThreadOptions>
Specifies how threads are created and used when a command runs in the session. The acceptable values for this parameter are:

- Default

- ReuseThread

- UseCurrentThread

- UseNewThread

The default value is UseCurrentThread .

For more information, see PSThreadOptions Enumeration (/dotnet/api/system.management.automation.runspaces.psthreadoptions?view=powershellsdk-1.1.0).

-TransportOption <System.Management.Automation.PSTransportOption>

Specifies the transport option.

This parameter was introduced in PowerShell 3.0.

-UseSharedProcess <System.Management.Automation.SwitchParameter>

Use only one process to host all sessions that are started by the same user and use the same session configuration. By default, each session is hosted in its own process.

This parameter was introduced in PowerShell 3.0.

-Confirm <System.Management.Automation.SwitchParameter>

Prompts you for confirmation before running the cmdlet.

-WhatIf <System.Management.Automation.SwitchParameter>

Shows what would happen if the cmdlet runs. The cmdlet is not run.

<CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see about_CommonParameters (https:/go.microsoft.com/fwlink/?LinkID=113216).

----- Example 1: Register a NewShell session configuration -----

```
$sessionConfiguration = @{
    Name='NewShell'
    ApplicationBase='c:\MyShells\'
    AssemblyName='MyShell.dll'
    ConfigurationTypeName='MyClass'
}
Register-PSSessionConfiguration @sessionConfiguration
```

To use this configuration, type `New-PSSession -ConfigurationName newshell`.

- Example 2: Register a MaintenanceShell session configuration -

```
Register-PSSessionConfiguration -Name MaintenanceShell -StartupScript
C:\ps-test\Maintenance.ps1
```

When a user uses a `New-PSSession` command and selects the MaintenanceShell configuration, the `Maintenance.ps1` script runs in the new session. The script can configure the session. This includes importing modules and setting the execution policy for the session. If the script generates any errors, including non-terminating errors, the `New-PSSession` command fails.

--------- Example 3: Register a session configuration ---------

```
$sddl = "O:NSG:BAD:P(A;;GA;;;BA)S:P(AU;FA;GA;;;WD)(AU;FASA;GWGX;;;WD)"
$sessionParams = @{
    Name="AdminShell"
    SecurityDescriptorSDDL=$sddl
    MaximumReceivedObjectSizeMB=20
    StartupScript="C:\scripts\AdminShell.ps1"
}
Register-PSSessionConfiguration @sessionParams
```

----- Example 4: Return a configuration container element -----

```
$s = Register-PSSessionConfiguration -Name MaintenanceShell -StartupScript
C:\ps-test\Maintenance.ps1
$s | Format-List -Property *
dir WSMan:\LocalHost\Plugin

PSPath           :
Microsoft.WSMan.Management\WSMan::localhost\Plugin\MaintenanceShell
PSParentPath     : Microsoft.WSMan.Management\WSMan::localhost\Plugin
PSChildName      : MaintenanceShell
PSDrive          : WSMan
PSProvider       : Microsoft.WSMan.Management\WSMan
PSIsContainer    : True
Keys             : {Name=MaintenanceShell}
Name             : MaintenanceShell
TypeNameOfElement : Container

Name                 Type            Keys
----                 ----            ----
MaintenanceShell     Container       {Name=MaintenanceShell}
microsoft.powershell Container       {Name=microsoft.powershell}
microsoft.powershell32 Container     {Name=microsoft.powershell32}
```

Example 5: Register a session configuration with a startup script

```
Register-PSSessionConfiguration -Name WithProfile -StartupScript
Add-Profile.ps1
```

The script contains a single command that uses dot sourcing to run the user's CurrentUserAllHosts profile in the current scope of the session.

For more information about profiles, see about_Profiles (./About/about_Profiles.md). For more information about dot sourcing, see

about_Scopes (./About/about_Scopes.md).

REMARKS

To see the examples, type: "get-help Register-PSSessionConfiguration -examples".

For more information, type: "get-help Register-PSSessionConfiguration -detailed".

For technical information, type: "get-help Register-PSSessionConfiguration -full".

For online help, type: "get-help Register-PSSessionConfiguration -online"