Windows PowerShell Get-Help on Cmdlet 'New-PSSessionConfigurationFile'

PS:\>Get-HELP New-PSSessionConfigurationFile -Full

NAME

New-PSSessionConfigurationFile

SYNOPSIS

Creates a file that defines a session configuration.

SYNTAX

New-PSSessionConfigurationFile [-Path] <System.String> [-AliasDefinitions <System.Collections.IDictionary[]>] [-AssembliesToLoad <System.String[]>] [-Author <System.String>] [-CompanyName <System.String>] [-Copyright <System.String>] [-Description <System.String>] [-EnvironmentVariables <System.Collections.IDictionary>] [-ExecutionPolicy {Unrestricted | RemoteSigned | AllSigned | Restricted | Default | Bypass | Undefined}] [-FormatsToProcess <System.String[]>] [-Full] [-FunctionDefinitions <System.Collections.IDictionary[]>] [-GroupManagedServiceAccount <System.String>] [-Guid <System.Guid>] [-LanguageMode {FullLanguage | RestrictedLanguage | NoLanguage | ConstrainedLanguage}] [-ModulesToImport <System.Object[]>] [-MountUserDrive] [-PowerShellVersion <System.Version>] [-RequiredGroups <System.Collections.IDictionary>] [-RoleDefinitions <System.Collections.IDictionary>] [-RunAsVirtualAccount] [-RunAsVirtualAccountGroups <System.String[]>]

[-SchemaVersion <System.Version>] [-ScriptsToProcess <System.String[]>] [-SessionType {Empty | RestrictedRemoteServer | Default}] [-TranscriptDirectory

<System.String>] [-TypesToProcess <System.String[]>] [-UserDriveMaximumSize <System.Int64>] [-VariableDefinitions <System.Object>] [-VisibleAliases <System.String[]>]

[-VisibleCmdlets <System.Object[]>] [-VisibleExternalCommands <System.String[]>] [-VisibleFunctions <System.Object[]>] [-VisibleProviders <System.String[]>]

[<CommonParameters>]

DESCRIPTION

The `New-PSSessionConfigurationFile` cmdlet creates a file of settings that define a session configuration and the environment of sessions that are created by using

the session configuration. To use the file in a session configuration, use the Path parameter of the `Register-PSSessionConfiguration` or `Set-PSSessionConfiguration`

cmdlets.

The session configuration file that `New-PSSessionConfigurationFile` creates is a human-readable text file that contains a hash table of the session configuration

properties and values. The file has a `.pssc` filename extension.

All parameters of `New-PSSessionConfigurationFile` are optional, except for the Path parameter. If you omit a parameter, the corresponding key in the session

configuration file is commented-out, except where noted in the parameter description.

A session configuration, also known as an endpoint, is a collection of settings on the local computer that define the environment for PowerShell sessions (PSSessions

) that connect to the computer. All PSSessions use a session configuration. To specify a particular session configuration, use the ConfigurationName parameter of

cmdlets that create a session, such as the `New-PSSession` cmdlet.

A session configuration file makes it easy to define a session configuration without complex scripts or code assemblies. The settings in the file are used with the

optional startup script and any assemblies in the session configuration.

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

about_Session_Configuration_Files (About/about_Session_Configuration_Files.md).

This cmdlet was introduced in PowerShell 3.0.

PARAMETERS

-AliasDefinitions <System.Collections.IDictionary[]>

Adds the specified aliases to sessions that use the session configuration. Enter a hash table with the following keys:

- Name - Name of the alias. This key is required.

- Value - The command that the alias represents. This key is required.

- Description - A text string that describes the alias. This key is optional.

- Options - Alias options. This key is optional. The default value is None . The acceptable values for this parameter are: None, ReadOnly, Constant, Private, or

AllScope.

For example: `@{Name='hlp';Value='Get-Help';Description='Gets help';Options='ReadOnly'}`

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-AssembliesToLoad <System.String[]>

Specifies the assemblies to load into the sessions that use the session configuration.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-Author <System.String>

Specifies the author of the session configuration or the configuration file. The default is the current user. The value of this parameter is visible in the

session configuration file, but it is not a property of the session configuration object.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-CompanyName <System.String>

Specifies the company that created the session configuration or the configuration file. The default value is Unknown . The value of this parameter is visible in

the session configuration file, but it is not a property of the session configuration object.

Required? false

Position? named

Default value Unknown

Accept pipeline input? False

Accept wildcard characters? false

-Copyright <System.String>

Specifies a copyright the session configuration file. The value of this parameter is visible in the session configuration file, but it is not a property of the

session configuration object.

If you omit this parameter, `New-PSSessionConfigurationFile` generates a copyright statement by using the value of the Author parameter.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-Description <System.String>

Specifies a description of the session configuration or the session configuration file. The value of this parameter is visible in the session configuration file,

but it is not a property of the session configuration object.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-EnvironmentVariables <System.Collections.IDictionary>

Adds environment variables to the session. Enter a hash table in which the keys are the environment variable names and the values are the environment variable

values.

For example: `EnvironmentVariables=@{TestShare='\Server01\TestShare'}`

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-ExecutionPolicy <Microsoft.PowerShell.ExecutionPolicy>

Specifies the execution policy of sessions that use the session configuration. If you omit this parameter, the value of the ExecutionPolicy key in the session

configuration file is Restricted . For information about execution policies in PowerShell, see about_Execution_Policies (about/about_Execution_Policies.md).

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-FormatsToProcess <System.String[]>

Specifies the formatting files (.ps1xml) that run in sessions that use the session configuration. The value of this parameter must be a full or absolute path of

the formatting files.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-Full <System.Management.Automation.SwitchParameter>

Indicates that this operation includes all possible configuration properties in the session configuration file.

Required? false

Position? named

Default value False

Accept pipeline input? False

Accept wildcard characters? false

-FunctionDefinitions <System.Collections.IDictionary[]>

Adds the specified functions to sessions that use the session configuration. Enter a hash table with the following keys:

- Name - Name of the function. This key is required.

- ScriptBlock - Function body. Enter a script block. This key is required.

- Options - Function options. This key is optional. The default value is None . The acceptable values for this parameter are: None, ReadOnly, Constant, Private,
or AllScope.

For example: `@{Name='Get-PowerShellProcess';ScriptBlock={Get-Process PowerShell};Options='AllScope'}`

Required? false
Position? named
Default value None
Accept pipeline input? False
Accept wildcard characters? false

-GroupManagedServiceAccount <System.String>

Configures sessions using this session configuration to run under the context of the specified Group Managed Service Account. The machine where this session
configuration is registered must have permission to request the gMSA password in order for sessions to be created successfully. This field cannot be used with the
RunAsVirtualAccount parameter.

Required? false
Position? named
Default value None
Accept pipeline input? False
Accept wildcard characters? false

-Guid <System.Guid>

Specifies a unique identifier for the session configuration file. If you omit this parameter,

`New-PSSessionConfigurationFile` generates a GUID for the file. To

create a new GUID in PowerShell, type `New-Guid`.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-LanguageMode <System.Management.Automation.PSLanguageMode>

Determines which elements of the PowerShell language are permitted in sessions that use this session configuration. You can use this parameter to restrict the

commands that particular users can run on the computer.

The acceptable values for this parameter are:

- FullLanguage - All language elements are permitted.

- ConstrainedLanguage - Commands that contain scripts to be evaluated are not allowed. The

ConstrainedLanguage mode restricts user access to Microsoft .NET Framework types, objects, or methods. - NoLanguage - Users may run cmdlets and functions, but

are not permitted to use any language elements, such as script blocks, variables, or operators. - RestrictedLanguage - Users may run cmdlets and functions, but

are not permitted to use script blocks or variables except for the following permitted variables: `$PSCulture`, `$PSUICulture`, `$True`, `$False`, and

`$Null`. Users may use only the basic comparison operators (`-eq`, `-gt`, `-lt`). Assignment statements, property references, and method calls are not permitted.

The default value of the LanguageMode parameter depends on the value of the SessionType parameter.

- Empty - NoLanguage

- RestrictedRemoteServer - NoLanguage

- Default - FullLanguage

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-ModulesToImport <System.Object[]>

Specifies the modules and snap-ins that are automatically imported into sessions that use the session configuration.

By default, only the Microsoft.PowerShell.Core snap-in is imported into remote sessions, but unless the cmdlets are excluded, users can use the `Import-Module`

and `Add-PSSnapin` cmdlets to add modules and snap-ins to the session.

Each module or snap-in in the value of this parameter can be represented by a string or as a hash table. A module string consists only of the name of the module

or snap-in. A module hash table can include ModuleName , ModuleVersion , and GUID keys. Only the ModuleName key is required.

For example, the following value consists of a string and a hash table. Any combination of strings and hash tables, in any order, is valid.

`'TroubleshootingPack', @{ModuleName='PSDiagnostics'; ModuleVersion='1.0.0.0';GUID='c61d6278-02a3-4618-ae37-a524d40a7f44'}`

The value of the ModulesToImport parameter of the `Register-PSSessionConfiguration` cmdlet takes precedence over the value of the ModulesToImport key in the

session configuration file.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-MountUserDrive <System.Management.Automation.SwitchParameter>

Configures sessions that use this session configuration to expose the `User:` PSDrive. User drives are unique for each connecting user and allow users to copy

data to and from PowerShell endpoints even if the File System provider is not exposed. User drive roots are created under

`$env:LOCALAPPDATA\Microsoft\Windows\PowerShell\DriveRoots`. For each user connecting to the endpoint, a folder is created with the name

`${env:USERDOMAIN}_${env:USERNAME}`. For computers in workgroups, the value of `$env:USERDOMAIN` is the hostname.

Contents in the user drive persist across user sessions and are not automatically removed. By default, users can only store up to 50MB of data in the user drive.

This can be customized with the UserDriveMaximumSize parameter.

Required? false

Position? named

Default value False

Accept pipeline input? False

Accept wildcard characters? false

-Path <System.String>

Specifies the path and filename of the session configuration file. The file must have a `.pssc` file name extension.

Required? true

Position? 0

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-PowerShellVersion <System.Version>

Specifies the version of the PowerShell engine in sessions that use the session configuration. The acceptable values for this parameter are: 2.0 and 3.0. If you

omit this parameter, the PowerShellVersion key is commented-out and newest version of PowerShell runs in the session.

The value of the PSVersion parameter of the `Register-PSSessionConfiguration` cmdlet takes precedence over the value of the PowerShellVersion key in the session

configuration file.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-RequiredGroups <System.Collections.IDictionary>

Specifies conditional access rules for users connecting to sessions that use this session configuration.

Enter a hashtable to compose your list of rules using only 1 key per hashtable, 'And' or 'Or', and set the value to an array of security group names or additional

hashtables.

Example requiring connecting users to be members of a single group: `@{ And = 'MyRequiredGroup' }`

Example requiring users to belong to group A, or both groups B and C, to access the endpoint: `@{ Or = 'GroupA', @{ And = 'GroupB', 'GroupC' } }`

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-RoleDefinitions <System.Collections.IDictionary>

Specifies the mapping between security groups (or users) and role capabilities. Users will be granted access to all role capabilities which apply to their group

membership at the time the session is created.

Enter a hash table in which the keys are the name of the security group and the values are hash tables that contain a list of role capabilities that should be

made available to the security group.

For example: `@{'Contoso\Level 2 Helpdesk Users' = @{ RoleCapabilities = 'Maintenance', 'ADHelpDesk' }}`

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-RunAsVirtualAccount <System.Management.Automation.SwitchParameter>

Configures sessions using this session configuration to be run as the computer's (virtual) administrator account. This field cannot be used with the

GroupManagedServiceAccount parameter.

Required? false

Position? named

Default value False

Accept pipeline input? False

Accept wildcard characters? false

-RunAsVirtualAccountGroups <System.String[]>

Specifies the security groups to be associated with the virtual account when a session that uses the session configuration is run as a virtual account. If

omitted, the virtual account belongs to Domain Admins on domain controllers and Administrators on all other computers.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-SchemaVersion <System.Version>

Specifies the version of the session configuration file schema. The default value is "1.0.0.0".

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-ScriptsToProcess <System.String[]>

Adds the specified scripts to sessions that use the session configuration. Enter the path and file names of the scripts. The value of this parameter must be a

full or absolute path of script file names.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-SessionType <System.Management.Automation.Remoting.SessionType>

Specifies the type of session that is created by using the session configuration. The default value is Default. 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.

This option is designed for you to create custom sessions by adding selected commands. If you do not add commands to an empty session, the session is limited

to expressions and might not be usable. - Default - Adds the Microsoft.PowerShell.Core module to the session. This module includes the `Import-Module` cmdlet

that users can use to import other modules unless you explicitly prohibit this cmdlet. - RestrictedRemoteServer. Includes only the following proxy functions:

`Exit-PSSession`, `Get-Command`, `Get-FormatData`, `Get-Help`, `Measure-Object`, `Out-Default`, and `Select-Object`. Use the parameters of this cmdlet to add

modules, functions, scripts, and other features to the session.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-TranscriptDirectory <System.String>

Specifies the directory to place session transcripts for sessions using this session configuration.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-TypesToProcess <System.String[]>

Adds the specified `.ps1xml` type files to sessions that use the session configuration. Enter the type filenames. The value of this parameter must be a full or

absolute path to type filenames.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-UserDriveMaximumSize <System.Int64>

Specifies the maximum size for user drives exposed in sessions that use this session configuration. When omitted, the default size of each `User:` drive root is

50MB.

This parameter should be used with MountUserDrive .

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-VariableDefinitions <System.Object>

Adds the specified variables to sessions that use the session configuration. Enter a hash table with the following keys:

- Name - Name of the variable. This key is required.

- Value - Variable value. This key is required.

For example: `@{Name='WarningPreference';Value='SilentlyContinue'}`

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-VisibleAliases <System.String[]>

Limits the aliases in the session to those specified in the value of this parameter, plus any aliases that you define in the AliasDefinition parameter. Wildcard

characters are supported. By default, all aliases that are defined by the PowerShell engine and all aliases that modules export are visible in the session.

For example: `VisibleAliases='gcm', 'gp'`

When any Visible parameter is included in the session configuration file, PowerShell removes the `Import-Module` cmdlet and its ipmo alias from the session.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? true

-VisibleCmdlets <System.Object[]>

Limits the cmdlets in the session to those specified in the value of this parameter. Wildcard characters and Module Qualified Names are supported.

By default, all cmdlets that modules in the session export are visible in the session. Use the SessionType and ModulesToImport parameters to determine which

modules and snap-ins are imported into the session. If no modules in ModulesToImport expose the cmdlet, the appropriate module will attempt to be autoloaded.

When any Visible parameter is included in the session configuration file, PowerShell removes the `Import-Module` cmdlet and its ipmo alias from the session.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? false

-VisibleExternalCommands <System.String[]>

Limits the external binaries, scripts, and commands that can be executed in the session to those specified in the value of this parameter. Wildcard characters are

supported.

By default, no external commands are visible in the session.

When any Visible parameter is included in the session configuration file, PowerShell, removes the `Import-Module` cmdlet and its ipmo alias from the session.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? true

-VisibleFunctions <System.Object[]>

Limits the functions in the session to those specified in the value of this parameter, plus any functions that you define in the FunctionDefinition parameter.

Wildcard characters are supported.

By default, all functions that modules in the session export are visible in the session. Use the SessionType and ModulesToImport parameters to determine which

modules and snap-ins are imported into the session.

When any Visible parameter is included in the session configuration file, PowerShell removes the `Import-Module` cmdlet and its ipmo alias from the session.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? true

-VisibleProviders <System.String[]>

Limits the PowerShell providers in the session to those specified in the value of this parameter. Wildcard characters are supported.

By default, all providers that modules in the session export are visible in the session. Use the SessionType and ModulesToImport parameters to determine which

modules are imported into the session.

When any Visible parameter is included in the session configuration file, PowerShell removes the `Import-Module` cmdlet and its `ipmo` alias from the session.

Required? false

Position? named

Default value None

Accept pipeline input? False

Accept wildcard characters? true

<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).

INPUTS

None

You can't pipe objects to this cmdlet.

OUTPUTS

None

This cmdlet returns no output.

NOTES

Windows PowerShell includes the following aliases for `New-PSSessionConfigurationFile`:

- `npssc`

Parameters, such as VisibleCmdlets and VisibleProviders , do not import items into the session. Instead, they select from among the items imported into the

session. For example, if the value of the VisibleProviders parameter is the Certificate provider, but the ModulesToImport parameter does not specify the

Microsoft.PowerShell.Security module that contains the Certificate provider, the Certificate provider is not visible in the session.

`New-PSSessionConfigurationFile` creates a session configuration file that has a .pssc file name extension in the path that you specify in the Path parameter.

When you use the session configuration file to create a session configuration, the `Register-PSSessionConfiguration` cmdlet copies the configuration file and

saves an active copy of the file in the SessionConfig subdirectory of the `$PSHOME` directory.

The ConfigFilePath property of the session configuration contains the fully qualified path of the active session configuration file. You can modify the active

configuration file in the `$PSHOME` directory at any time using any text editor. The changes that you make affect all new sessions that use the session

configuration, but not existing sessions.

Before using an edited session configuration file, use the `Test-PSSessionConfigurationFile` cmdlet to verify that the configuration file entries are valid.

------ Example 1: Creating and using a NoLanguage session ------

```
New-PSSessionConfigurationFile -Path .\NoLanguage.pssc -LanguageMode NoLanguage
Register-PSSessionConfiguration -Path .\NoLanguage.pssc -Name NoLanguage -Force
$NoLanguageSession = New-PSSession -ComputerName Srv01 -ConfigurationName NoLanguage
Invoke-Command -Session $NoLanguageSession -ScriptBlock {
  if ((Get-Date) -lt '1January2099') {'Before'} else {'After'}
}
```

```
The syntax is not supported by this runspace. This might be because it is in no-language mode.
    + CategoryInfo          : ParserError: (if ((Get-Date) ...') {'Before'}  :String) [], ParseException
    + FullyQualifiedErrorId : ScriptsNotAllowed
    + PSComputerName        : localhost
```

In this example, the `Invoke-Command` fails because the LanguageMode is set to NoLanguage .

-- Example 2: Creating and using a RestrictedLanguage session --

```
New-PSSessionConfigurationFile -Path .\NoLanguage.pssc -LanguageMode RestrictedLanguage
Register-PSSessionConfiguration -Path .\NoLanguage.pssc -Name RestrictedLanguage -Force
$RestrictedSession = New-PSSession -ComputerName Srv01 -ConfigurationName RestrictedLanguage
Invoke-Command -Session $RestrictedSession -ScriptBlock {
  if ((Get-Date) -lt '1January2099') {'Before'} else {'After'}
}
```

```
Before
```

In this example, the `Invoke-Command` succeeds because the LanguageMode is set to RestrictedLanguage .

------- Example 3: Changing a Session Configuration File -------

```
New-PSSessionConfigurationFile -Path .\New-ITTasks.pssc -ModulesToImport Microsoft*, ITTasks, PSScheduledJob
Set-PSSessionConfiguration -Name ITTasks -Path .\New-ITTasks.pssc
```

The `New-PSSessionConfigurationFile` cmdlet to create a session configuration file that imports the required modules. The `Set-PSSessionConfiguration` cmdlet replaces

the current configuration file with the new one. This new configuration only affects new sessions created after the change. Existing "ITTasks" sessions are not

affected.

------- Example 4: Editing a Session Configuration File -------

```
$ITConfig = Get-PSSessionConfiguration -Name ITConfig
notepad.exe $ITConfig.ConfigFilePath
Test-PSSessionConfigurationFile -Path $ITConfig.ConfigFilePath

True
```

Use the Verbose parameter with `Test-PSSessionConfigurationFile` to display any errors that are detected. The cmdlet returns `$True` if no errors are detected in the

file.

-------- Example 5: Create a sample configuration file --------

```
$configSettings = @{
    Path = '.\SampleFile.pssc'
    SchemaVersion = '1.0.0.0'
    Author = 'User01'
    Copyright = '(c) Fabrikam Corporation. All rights reserved.'
    CompanyName = 'Fabrikam Corporation'
    Description = 'This is a sample file.'
    ExecutionPolicy = 'AllSigned'
    PowerShellVersion = '3.0'
    LanguageMode = 'FullLanguage'
    SessionType = 'Default'
    EnvironmentVariables = @{TESTSHARE='\\Test2\Test'}
                                ModulesToImport = @{ModuleName='PSScheduledJob'; ModuleVersion='1.0.0.0';
GUID='50cdb55f-5ab7-489f-9e94-4ec21ff51e59'},'PSDiagnostics'
    AssembliesToLoad = 'System.Web.Services','FSharp.Compiler.CodeDom.dll'
```

```
    TypesToProcess = 'Types1.ps1xml','Types2.ps1xml'
    FormatsToProcess = 'CustomFormats.ps1xml'
    ScriptsToProcess = 'Get-Inputs.ps1'
    AliasDefinitions = @{Name='hlp';Value='Get-Help';Description='Gets help.';Options='AllScope'},
      @{Name='Update';Value='Update-Help';Description='Updates help';Options='ReadOnly'}
                        FunctionDefinitions = @{Name='Get-Function';ScriptBlock={Get-Command -CommandType
Function};Options='ReadOnly'}
    VariableDefinitions = @{Name='WarningPreference';Value='SilentlyContinue'}
    VisibleAliases = 'c*','g*','i*','s*'
    VisibleCmdlets = 'Get*'
    VisibleFunctions = 'Get*'
    VisibleProviders = 'FileSystem','Function','Variable'
    RunAsVirtualAccount = $true
    RunAsVirtualAccountGroups = 'Backup Operators'
  }
  New-PSSessionConfigurationFile @configSettings
  Get-Content SampleFile.pssc

  @{

  # Version number of the schema used for this document
  SchemaVersion = '1.0.0.0'

  # ID used to uniquely identify this document
  GUID = '1caeff7f-27ca-4360-97cf-37846f594235'

  # Author of this document
  Author = 'User01'

  # Description of the functionality provided by these settings
  Description = 'This is a sample file.'

  # Company associated with this document
```

```
    CompanyName = 'Fabrikam Corporation'

    # Copyright statement for this document
    Copyright = '(c) Fabrikam Corporation. All rights reserved.'

    # Session type defaults to apply for this session configuration. Can be 'RestrictedRemoteServer' (recommended), 'Empty',
or 'Default'
    SessionType = 'Default'

    # Directory to place session transcripts for this session configuration
    # TranscriptDirectory = 'C:\Transcripts\'

    # Whether to run this session configuration as the machine's (virtual) administrator account
    RunAsVirtualAccount = $true

    # Groups associated with machine's (virtual) administrator account
    RunAsVirtualAccountGroups = 'Backup Operators'

    # Scripts to run when applied to a session
    ScriptsToProcess = 'Get-Inputs.ps1'

    # User roles (security groups), and the role capabilities that should be applied to them when applied to a session
    # RoleDefinitions = @{ 'CONTOSO\SqlAdmins' = @{ RoleCapabilities = 'SqlAdministration' }; 'CONTOSO\SqlManaged' =
@{ RoleCapabilityFiles =
    'C:\RoleCapability\SqlManaged.psrc' }; 'CONTOSO\ServerMonitors' = @{ VisibleCmdlets = 'Get-Process' } }

     # Language mode to apply when applied to a session. Can be 'NoLanguage' (recommended), 'RestrictedLanguage',
'ConstrainedLanguage', or 'FullLanguage'
    LanguageMode = 'FullLanguage'

    # Execution policy to apply when applied to a session
    ExecutionPolicy = 'AllSigned'
```

```
# Version of the PowerShell engine to use when applied to a session
PowerShellVersion = '3.0'

# Modules to import when applied to a session
ModulesToImport = @{
    'GUID' = '50cdb55f-5ab7-489f-9e94-4ec21ff51e59'
    'ModuleName' = 'PSScheduledJob'
    'ModuleVersion' = '1.0.0.0' }, 'PSDiagnostics'

# Aliases to make visible when applied to a session
VisibleAliases = 'c*', 'g*', 'i*', 's*'

# Cmdlets to make visible when applied to a session
VisibleCmdlets = 'Get*'

# Functions to make visible when applied to a session
VisibleFunctions = 'Get*'

# Providers to make visible when applied to a session
VisibleProviders = 'FileSystem', 'Function', 'Variable'

# Aliases to be defined when applied to a session
AliasDefinitions = @{
    'Description' = 'Gets help.'
    'Name' = 'hlp'
    'Options' = 'AllScope'
    'Value' = 'Get-Help' }, @{
    'Description' = 'Updates help'
    'Name' = 'Update'
    'Options' = 'ReadOnly'
    'Value' = 'Update-Help' }

# Functions to define when applied to a session
```

```
    FunctionDefinitions = @{
        'Name' = 'Get-Function'
        'Options' = 'ReadOnly'
        'ScriptBlock' = {Get-Command -CommandType Function} }

    # Variables to define when applied to a session
    VariableDefinitions = @{
        'Name' = 'WarningPreference'
        'Value' = 'SilentlyContinue' }

    # Environment variables to define when applied to a session
    EnvironmentVariables = @{
        'TESTSHARE' = '\\Test2\Test' }

    # Type files (.ps1xml) to load when applied to a session
    TypesToProcess = 'Types1.ps1xml', 'Types2.ps1xml'

    # Format files (.ps1xml) to load when applied to a session
    FormatsToProcess = 'CustomFormats.ps1xml'

    # Assemblies to load when applied to a session
    AssembliesToLoad = 'System.Web.Services', 'FSharp.Compiler.CodeDom.dll'

    }
```

RELATED LINKS

Online Version:

https://learn.microsoft.com/powershell/module/microsoft.powershell.core/new-pssessionconfigurationfile?view=powershell-5.1&WT.mc_id=ps-gethelp

Disable-PSSessionConfiguration

Enable-PSSessionConfiguration

Get-PSSessionConfiguration

Register-PSSessionConfiguration

Set-PSSessionConfiguration

Test-PSSessionConfigurationFile

Unregister-PSSessionConfiguration

WSMan Provider

about_Session_Configurations

about_Session_Configuration_Files