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

PowerShell Get-Help on command 'Out-String'

PS C:\Users\wahid> Get-Help Out-String

NAME

Out-String

SYNOPSIS

Outputs input objects as a string.

SYNTAX

Out-String [-InputObject <System.Management.Automation.PSObject>] [-Stream]

[-Width <System.Int32>] [<CommonParameters>]

DESCRIPTION

The `Out-String` cmdlet converts input objects into strings. By default,

`Out-String` accumulates the strings and returns them as a single string, but

you can use the Stream parameter to direct `Out-String` to return one line at

a time or create an array of strings. This cmdlet lets you search and

manipulate string output as you would in traditional shells when object

manipulation is less convenient.

PowerShell also adds the `OSS` function that calls `Out-String -Stream` as a

shorthand way to use `Out-String` in a pipeline.

PARAMETERS

-InputObject <System.Management.Automation.PSObject>

Specifies the objects to be written to a string. Enter a variable that contains the objects, or type a command or expression that gets the objects.

-Stream <System.Management.Automation.SwitchParameter>

By default, `Out-String` outputs a single string formatted as you would see it in the console including any blank headers or trailing newlines. The Stream parameter enables `Out-String` to output each line one by one. The only exception to this are multiline strings. In that case, `Out-String` will still output the string as a single, multiline string.

-Width <System.Int32>

Specifies the number of characters in each line of output. Any additional characters are wrapped to the next line or truncated depending on the formatter cmdlet used. The Width parameter applies only to objects that are being formatted. If you omit this parameter, the width is determined by the characteristics of the host program. In terminal (console) windows, the current window width is used as the default value. PowerShell console windows default to a width of 80 characters on installation.

<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: Get the current culture and convert the data to strings

```
$C = Get-Culture | Select-Object -Property *
Out-String -InputObject $C -Width 100

Parent                    : en
LCID                      : 1033
KeyboardLayoutId          : 1033
Name                      : en-US
IetfLanguageTag           : en-US
DisplayName               : English (United States)
NativeName                : English (United States)
EnglishName               : English (United States)
TwoLetterISOLanguageName  : en
ThreeLetterISOLanguageName : eng
ThreeLetterWindowsLanguageName : ENU
CompareInfo               : CompareInfo - en-US
TextInfo                  : TextInfo - en-US
IsNeutralCulture          : False
CultureTypes              : SpecificCultures, InstalledWin32Cultures,
FrameworkCultures
NumberFormat              : System.Globalization.NumberFormatInfo
DateTimeFormat            : System.Globalization.DateTimeFormatInfo
Calendar                  : System.Globalization.GregorianCalendar
OptionalCalendars         : {System.Globalization.GregorianCalendar,
                            System.Globalization.GregorianCalendar}
UseUserOverride           : True
IsReadOnly                : False
```

The `$C` variable stores a Selected.System.Globalization.CultureInfo object. The object is the result of `Get-Culture` sending output down the pipeline to `Select-Object`. The Property parameter uses an asterisk (`*`) wildcard to specify all properties are contained in the object.

`Out-String` uses the InputObject parameter to specify the CultureInfo object

stored in the `$C` variable. The objects in `$C` are converted to a string.

> [!NOTE] > To view the `Out-String` array, store the output to a variable and use an array index to view the > elements. For more information about the array index, see > about_Arrays (../microsoft.powershell.core/about/about_arrays.md). > > `$str = Out-String -InputObject $C -Width 100`

---------- Example 2: Working with objects ----------

```
Get-Alias | Out-String -Stream | Select-String -Pattern "gcm"

Alias           gcm -> Get-Command
```

`Get-Alias` gets the System.Management.Automation.AliasInfo objects, one for each alias, and sends the objects down the pipeline. `Out-String` uses the Stream parameter to convert each object to a string rather than concatenating all the objects into a single string. The System.String objects are sent down the pipeline and `Select-String` uses the Pattern parameter to find matches for the text gcm .

> [!NOTE] > If you omit the Stream parameter, the command displays all the aliases because `Select-String` > finds the text gcm in the single string that `Out-String` returns.

-- Example 3: Use the Width parameter to prevent truncation. --

```
PS> @{TestKey = ('x' * 200)} | Out-String
Name                           Value
----                           -----
TestKey                        xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx...

PS> @{TestKey = ('x' * 200)} | Out-String -Width 250

Name                           Value
```

----   -----

TestKey

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

xxxxxxxxxxxxxxxxxxxxx

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

REMARKS

To see the examples, type: "get-help Out-String -examples".

For more information, type: "get-help Out-String -detailed".

For technical information, type: "get-help Out-String -full".

For online help, type: "get-help Out-String -online"