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

PowerShell Get-Help on command 'Out-File'

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

NAME

Out-File

SYNOPSIS

Sends output to a file.

SYNTAX

Out-File [-FilePath] <System.String> [[-Encoding] {ASCII | BigEndianUnicode | Default | OEM | String | Unicode | Unknown | UTF7 | UTF8 | UTF32}] [-Append] [-Force] [-InputObject <System.Management.Automation.PSObject>] [-NoClobber] [-NoNewline] [-Width <System.Int32>] [-Confirm] [-WhatIf] [<CommonParameters>]

Out-File [[-Encoding] {ASCII | BigEndianUnicode | Default | OEM | String | Unicode | Unknown | UTF7 | UTF8 | UTF32}] [-Append] [-Force] [-InputObject <System.Management.Automation.PSObject>] -LiteralPath <System.String> [-NoClobber] [-NoNewline] [-Width <System.Int32>] [-Confirm] [-WhatIf] [<CommonParameters>]

DESCRIPTION

The `Out-File` cmdlet sends output to a file. It implicitly uses PowerShell's formatting system to write to the file. The file receives the same display representation as the terminal. This means that the output may not be ideal for programmatic processing unless all input objects are strings. When you need to specify parameters for the output, use `Out-File` rather than the redirection operator (`>`). For more information about redirection, see about_Redirection (../Microsoft.PowerShell.Core/About/about_Redirection.md).

PARAMETERS

-Append <System.Management.Automation.SwitchParameter>

Adds the output to the end of an existing file. If no Encoding is specified, the cmdlet uses the default encoding. That encoding may not match the encoding of the target file. This is the same behavior as the redirection operator (`>>`).

-Encoding <System.String>

Specifies the type of encoding for the target file. The default value is `unicode`.

The acceptable values for this parameter are as follows:

- `ascii` Uses ASCII (7-bit) character set.

- `bigendianunicode` Uses UTF-16 with the big-endian byte order.

- `default` Uses the encoding that corresponds to the system's active code page (usually ANSI).

- `oem` Uses the encoding that corresponds to the system's current OEM code page.

- `string` Same as `unicode`.

- `unicode` Uses UTF-16 with the little-endian byte order.

- `unknown` Same as `unicode`.

- `utf7` Uses UTF-7.

- `utf8` Uses UTF-8.

- `utf32` Uses UTF-32 with the little-endian byte order.

-FilePath <System.String>
Specifies the path to the output file.

-Force <System.Management.Automation.SwitchParameter>
Overrides the read-only attribute and overwrites an existing read-only file. The Force parameter doesn't override security restrictions.

-InputObject <System.Management.Automation.PSObject>
Specifies the objects to be written to the file. Enter a variable that contains the objects or type a command or expression that gets the objects.

-LiteralPath <System.String>
Specifies the path to the output file. The LiteralPath parameter is used exactly as it's typed. Wildcard characters aren't accepted. If the path includes escape characters, enclose it in single quotation marks. Single quotation marks tell PowerShell not to interpret any characters as escape sequences. For more information, see about_Quoting_Rules (../Microsoft.Powershell.Core/About/about_Quoting_Rules.md).

-NoClobber <System.Management.Automation.SwitchParameter>
NoClobber prevents an existing file from being overwritten and displays a message that the file already exists. By default, if a file exists in the

specified path, `Out-File` overwrites the file without warning.

-NoNewline <System.Management.Automation.SwitchParameter>
Specifies that the content written to the file doesn't end with a newline character. The string representations of the input objects are concatenated to form the output. No spaces or newlines are inserted between the output strings. No newline is added after the last output string.

-Width <System.Int32>
Specifies the maximum number of characters in each line of output. Any additional characters are truncated, not wrapped. If this parameter isn't used, the width is determined by the characteristics of the host. The default for the PowerShell console is 80 characters. If you want to control the width for all invocations of `Out-File` as well as the redirection operators (`>` and `>>`), set `$PSDefaultParameterValues['out-file:width'] = 2000` before using `Out-File`.

-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 isn't 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: Send output and create a file ----------

```
Get-Process | Out-File -FilePath .\Process.txt
Get-Content -Path .\Process.txt

NPM(K)    PM(M)      WS(M)     CPU(s)      Id  SI ProcessName
 ------    -----      -----     ------      --  -- -----------
     29    22.39      35.40      10.98   42764   9 Application
     53    99.04     113.96       0.00   32664   0 CcmExec
     27    96.62     112.43     113.00   17720   9 Code
```

The `Get-Process` cmdlet gets the list of processes running on the local computer. The Process objects are sent down the pipeline to the `Out-File` cmdlet. `Out-File` uses the FilePath parameter and creates a file in the current directory named Process.txt . The `Get-Content` command gets content from the file and displays it in the PowerShell console.

-- Example 2: Prevent an existing file from being overwritten --

```
Get-Process | Out-File -FilePath .\Process.txt -NoClobber

Out-File : The file 'C:\Test\Process.txt' already exists.
At line:1 char:15
+ Get-Process | Out-File -FilePath .\Process.txt -NoClobber
+               ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

The `Get-Process` cmdlet gets the list of processes running on the local computer. The Process objects are sent down the pipeline to the `Out-File` cmdlet. `Out-File` uses the FilePath parameter and attempts to write to a file in the current directory named Process.txt . The NoClobber parameter prevents the file from being overwritten and displays a message that the file already exists.

------- Example 3: Send output to a file in ASCII format -------

```
$Procs = Get-Process
Out-File -FilePath .\Process.txt -InputObject $Procs -Encoding ASCII -Width 50
```

The `Get-Process` cmdlet gets the list of processes running on the local computer. The Process objects are stored in the variable, `$Procs`. `Out-File` uses the FilePath parameter and creates a file in the current directory named Process.txt . The InputObject parameter passes the process objects in `$Procs` to the file Process.txt . The Encoding parameter converts the output to ASCII format. The Width parameter limits each line in the file to 50 characters so some data might be truncated.

----- Example 4: Use a provider and send output to a file -----

```
PS> Set-Location -Path Alias:

PS> Get-Location

Path
----
Alias:\

PS> Get-ChildItem | Out-File -FilePath C:\TestDir\AliasNames.txt

PS> Get-Content -Path C:\TestDir\AliasNames.txt

CommandType     Name
-----------     ----
Alias           % -> ForEach-Object
Alias           ? -> Where-Object
Alias           ac -> Add-Content
Alias           cat -> Get-Content
```

The `Set-Location` command uses the Path parameter to set the current location to the registry provider `Alias:`. The `Get-Location` cmdlet displays the complete path for `Alias:`. `Get-ChildItem` sends objects down the pipeline to the `Out-File` cmdlet. `Out-File` uses the FilePath parameter to specify the

```
    complete path and filename for the output, C:\TestDir\AliasNames.txt . The
    `Get-Content` cmdlet uses the Path parameter and displays the file's content
    in the PowerShell console.
    ------ Example 5: Set file output width for entire scope ------

    function DemoDefaultOutFileWidth() {
        try {
            $PSDefaultParameterValues['out-file:width'] = 2000

            $logFile = "$pwd\logfile.txt"

            Get-ChildItem Env:\ > $logFile

            Get-Service -ErrorAction Ignore |
                Format-Table -AutoSize |
                Out-File $logFile -Append

            Get-Process | Format-Table Id,SI,Name,Path,MainWindowTitle >> $logFile
        }
        finally {
            $PSDefaultParameterValues.Remove('out-file:width')
        }
    }

    DemoDefaultOutFileWidth

    For more information about `$PSDefaultParameterValues`, see
    about_Preference_Variables (../Microsoft.Powershell.Core/About/about_preference
    _variables.md#psdefaultparametervalues).
REMARKS
    To see the examples, type: "get-help Out-File -examples".
    For more information, type: "get-help Out-File -detailed".
    For technical information, type: "get-help Out-File -full".
```

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