



python



PowerShell

FPDF Library
PDF generator

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

PowerShell Get-Help on command 'Export-FormatData'

PS C:\Users\wahid> Get-Help Export-FormatData

NAME

Export-FormatData

SYNOPSIS

Saves formatting data from the current session in a formatting file.

SYNTAX

```
Export-FormatData [-Force] [-IncludeScriptBlock] -InputObject  
<System.Management.Automation.ExtendedTypeDefinition[]> -LiteralPath  
<System.String> [-NoClobber] [<CommonParameters>]
```

```
Export-FormatData [-Force] [-IncludeScriptBlock] -InputObject  
<System.Management.Automation.ExtendedTypeDefinition[]> [-NoClobber] -Path  
<System.String> [<CommonParameters>]
```

DESCRIPTION

The `Export-FormatData` cmdlet creates Windows PowerShell formatting files (format.ps1xml) from the formatting objects in the current session. It takes the ExtendedTypeDefinition objects that `Get-FormatData` returns and saves

them in a file in XML format.

Windows PowerShell uses the data in formatting files (format.ps1xml) to generate the default display of Microsoft .NET Framework objects in the session. You can view and edit the formatting files and use the Update-FormatData cmdlet to add the formatting data to a session.

For more information about formatting files in Windows PowerShell, see `about_Format.ps1xml` (`../Microsoft.PowerShell.Core/About/about_Format.ps1xml.md`).

PARAMETERS

`-Force <System.Management.Automation.SwitchParameter>`

Forces the command to run without asking for user confirmation.

`-IncludeScriptBlock <System.Management.Automation.SwitchParameter>`

Indicates whether script blocks in the format data are exported.

Because script blocks contain code and can be used maliciously, they are not exported by default.

`-InputObject <System.Management.Automation.ExtendedTypeDefinition[]>`

Specifies the format data objects to be exported. Enter a variable that contains the objects or a command that gets the objects, such as a ``Get-FormatData`` command. You can also pipe the objects from ``Get-FormatData`` to ``Export-FormatData``.

`-LiteralPath <System.String>`

Specifies a location for the output file. Unlike the Path parameter, the value of LiteralPath is used exactly as it is typed. No characters are interpreted as wildcards. If the path includes escape characters, enclose it in single quotation marks. Single quotation marks tell Windows

PowerShell not to interpret any characters as escape sequences.

`-NoClobber <System.Management.Automation.SwitchParameter>`

Indicates that the cmdlet does not overwrite existing files. By default, ``Export-FormatData`` overwrites files without warning unless the file has the read-only attribute.

To direct ``Export-FormatData`` to overwrite read-only files, use the `Force` parameter.

`-Path <System.String>`

Specifies a location for the output file. Enter a path (optional) and file name with a `format.ps1xml` file name extension. If you omit the path, ``Export-FormatData`` creates the file in the current directory.

If you use a file name extension other than `.ps1xml`, the ``Update-FormatData`` cmdlet will not recognize the file.

If you specify an existing file, ``Export-FormatData`` overwrites the file without warning, unless the file has the read-only attribute. To overwrite a read-only file, use the `Force` parameter. To prevent files from being overwritten, use the `NoClobber` parameter.

`<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: Export session format data -----

```
Get-FormatData -TypeName "*" | Export-FormatData -Path "allformat.ps1xml"
```

```
-IncludeScriptBlock
```

This command exports all of the format data in the session to the AllFormat.ps1xml file.

The command uses the `Get-FormatData` cmdlet to get the format data in the session. A value of `*` (all) for the TypeName parameter directs the cmdlet to get all of the data in the session.`

The command uses a pipeline operator (``|`)` to send the format data from the `Get-FormatData` command to the Export-FormatData` cmdlet, which exports the format data to the AllFormat.ps1 file.`

The `Export-FormatData` command uses the IncludeScriptBlock parameter to include script blocks in the format data in the file.`

----- Example 2: Export format data for a type -----

```
$F = Get-FormatData -TypeName "helpinfoshort"
Export-FormatData -InputObject $F -Path "c:\test\help.format.ps1xml"
-IncludeScriptBlock
```

These commands export the format data for the HelpInfoShort type to the Help.format.ps1xml file.

The first command uses the `Get-FormatData` cmdlet to get the format data for the HelpInfoShort type, and it saves it in the `$F` variable.`

The second command uses the InputObject parameter of the `Export-FormatData` cmdlet to enter the format data saved in the `$F` variable. It also uses the IncludeScriptBlock parameter to include script blocks in the output.`

----- Example 3: Export format data without a script block -----

```
Get-FormatData -TypeName "System.Diagnostics.Process" | Export-FormatData
-Path process.format.ps1xml
```

```
Update-FormatData -PrependPath ".\process.format.ps1xml"
```

```
Get-Process p*
```

Handles	NPM(K)	PM(K)	WS(K)	VM(M)	CPU(s)	Id	ProcessName
-----	-----	-----	-----	-----	-----		
323			5600				powershell
336			3900				powershell_ise
138			4076				PresentationFontCache

This example shows the effect of omitting the `IncludeScriptBlock` parameter from an `Export-FormatData` command.

The first command uses the `Get-FormatData` cmdlet to get the format data for the `System.Diagnostics.Process` object that the `Get-Process` cmdlet returns. The command uses a pipeline operator (`|`) to send the formatting data to the `Export-FormatData` cmdlet, which exports it to the `Process.format.ps1xml` file in the current directory.

In this case, the `Export-FormatData` command does not use the `IncludeScriptBlock` parameter.

The second command uses the `Update-FormatData` cmdlet to add the `Process.format.ps1xml` file to the current session. The command uses the `PrependPath` parameter to ensure that the formatting data for process objects in the `Process.format.ps1xml` file is found before the standard formatting data for process objects.

The third command shows the effects of this change. The command uses the `Get-Process` cmdlet to get processes that have names that begin with P. The output shows that property values that are calculated by using script blocks are missing from the display.

REMARKS

To see the examples, type: "get-help Export-FormatData -examples".

For more information, type: "get-help Export-FormatData -detailed".

For technical information, type: "get-help Export-FormatData -full".

For online help, type: "get-help Export-FormatData -online"