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

PowerShell Get-Help on command 'Get-FormatData'

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

NAME

Get-FormatData

SYNOPSIS

Gets the formatting data in the current session.

SYNTAX

Get-FormatData [[-TypeName] <System.String[]>] [-PowerShellVersion <System.Version>] [<CommonParameters>]

DESCRIPTION

The `Get-FormatData` cmdlet gets the formatting data in the current session.

The formatting data in the session includes formatting data from `Format.ps1xml` formatting files, such as those in the `$PSHOME` directory, formatting data for modules that you import into the session, and formatting data for commands that you import into your session by using the `Import-PSSession` cmdlet.

You can use this cmdlet to examine the formatting data. Then, you can use the `Export-FormatData` cmdlet to serialize the objects, convert them to XML, and save them in `Format.ps1xml` files.

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

PARAMETERS

-PowerShellVersion <System.Version>

Specify the version of PowerShell this cmdlet gets for the formatting data. Enter a two digit number separated by a period.

This parameter was added in PowerShell 5.1 to improve compatibility when remoting computers running older versions of PowerShell.

-TypeName <System.String[]>

Specifies the type names that this cmdlet gets for the formatting data. Enter the type names. Wildcards are permitted.

<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 all formatting data ----------

Get-FormatData

--------- Example 2: Get formatting data by type name ---------

```
Get-FormatData -TypeName 'System.Management.Automation.Cmd*'



--------- Example 3: Examine a formatting data object ---------

$F = Get-FormatData -TypeName 'System.Management.Automation.Cmd*'
$F

TypeName        FormatViewDefinition
--------        --------------------
HelpInfoShort   {help , TableControl}

$F.FormatViewDefinition[0].control

Headers         : {System.Management.Automation.TableControlColumnHeader,
                  System.Management.Automation.TableControlColumnHeader,
                  System.Management.Automation.TableControlColumnHeader,
                  System.Management.Automation.TableControlColumnHeader}
Rows            : {System.Management.Automation.TableControlRow}
AutoSize        : False
HideTableHeaders : False
GroupBy         :
OutOfBand       : False

$F.FormatViewDefinition[0].control.Headers

Label       Alignment Width
-----       --------- -----
CommandType Undefined    15
Name        Undefined    50
Version     Undefined    10
Source      Undefined     0
```

--------- Example 4: Get formatting data and export it ---------

```
$A = Get-FormatData
Import-Module bitstransfer
$B = Get-FormatData
Compare-Object $A $B

InputObject                                    SideIndicator
-----------                                    -------------
Microsoft.BackgroundIntelligentTransfer.Management.BitsJob =>

Get-FormatData *bits* | Export-FormatData -FilePath c:\test\bits.format.ps1xml
Get-Content c:\test\bits.format.ps1xml

<?xml version="1.0" encoding="utf-8"?><Configuration><ViewDefinitions>
<View><Name>Microsoft.BackgroundIntelligentTransfer.Management.BitsJob</Name>
...
```

The first four commands use the `Get-FormatData`, `Import-Module`, and `Compare-Object` cmdlets to identify the format type that the BitsTransfer module adds to the session.

The fifth command uses the `Get-FormatData` cmdlet to get the format type that the BitsTransfer module adds. It uses a pipeline operator (`|`) to send the format type object to the `Export-FormatData` cmdlet, which converts it back to XML and saves it in the specified `format.ps1xml` file.

The final command shows an excerpt of the `format.ps1xml` file content.

Example 5: Get formatting data based on the specified version of PowerShell

```
Get-FormatData -TypeName 'Microsoft.Powershell.Utility.FileHash'
```

-PowerShellVersion $PSVersionTable.PSVersion

TypeNames FormatViewDefinition

--- ---

{Microsoft.Powershell.Utility.FileHash} {Microsoft.Powershell.Utility.FileHash}

> [!IMPORTANT] > To ensure that the complete type formatting information is returned, you should always include the > PowerShellVersion parameter with the appropriate version. If the parameter and value are > omitted, you may not get all the correct type information.

REMARKS

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

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

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

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