PowerShell Get-Help on command 'Group-Object'

PS C:\Users\wahid> Get-Help Group-Object

NAME

    Group-Object

SYNOPSIS

    Groups objects that contain the same value for specified properties.

SYNTAX

```
    Group-Object [[-Property] <System.Object[]>] [-AsHashTable] [-AsString]
    [-CaseSensitive] [-Culture <System.String>] [-InputObject
    <System.Management.Automation.PSObject>] [-NoElement] [<CommonParameters>]
```

DESCRIPTION

    The `Group-Object` cmdlet displays objects in groups based on the value of a
    specified property. `Group-Object` returns a table with one row for each
    property value and a column that displays the number of items with that value.

    If you specify more than one property, `Group-Object` first groups them by the
    values of the first property, and then, within each property group, it groups
    by the value of the next property.

PARAMETERS

-AsHashTable <System.Management.Automation.SwitchParameter>

Indicates that this cmdlet returns the group as a hash table. The keys of the hash table are the property values by which the objects are grouped. The values of the hash table are the objects that have that property value.

By itself, the AsHashTable parameter returns each hash table in which each key is an instance of the grouped object. When used with the AsString parameter, the keys in the hash table are strings.

-AsString <System.Management.Automation.SwitchParameter>

Indicates that this cmdlet converts the hash table keys to strings. By default, the hash table keys are instances of the grouped object. This parameter is valid only when used with the AsHashTable parameter.

-CaseSensitive <System.Management.Automation.SwitchParameter>

Indicates that this cmdlet makes the grouping case-sensitive. Without this parameter, the property values of objects in a group might have different cases.

-Culture <System.String>

Specifies the culture to use when comparing strings.

-InputObject <System.Management.Automation.PSObject>

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

When you use the InputObject parameter to submit a collection of objects to `Group-Object`, `Group-Object` receives one object that represents the collection. As a result, it creates a single group with that object as its member.

    To group the objects in a collection, pipe the objects to `Group-Object`.

-NoElement <System.Management.Automation.SwitchParameter>

    Indicates that this cmdlet omits the members of a group from the results.

-Property <System.Object[]>

    Specifies the properties for grouping. The objects are arranged into named groups based on the value of the specified properties. When no property is specified, objects are grouped by their value or the `ToString()` representation of their value. The output is presented in order the group objects were created.

    The value of the Property parameter can be a new calculated property. The calculated property can be a script block or a hash table. Valid key-value pairs are:

    - Expression - `<string>` or `<script block>`

    For more information, see about_Calculated_Properties (../Microsoft.PowerShell.Core/About/about_Calculated_Properties.md).

<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: Group files by extension -------------

```
$files = Get-ChildItem -Path $PSHOME -Recurse
$files |
    Group-Object -Property extension -NoElement |
    Sort-Object -Property Count -Descending

Count Name
----- ----
  365 .xml
  231 .cdxml
  197
  169 .ps1xml
  142 .txt
  114 .psd1
   63 .psm1
   49 .xsd
   36 .dll
   15 .mfl
   15 .mof
...
```

--------- Example 2: Group integers by odds and evens ---------

```
1..20 | Group-Object -Property {$_ % 2}

Count Name                      Group
----- ----                      -----
   10 0                         {2, 4, 6, 8...}
   10 1                         {1, 3, 5, 7...}
```

-------- Example 3: Group event log events by EntryType --------

```
Get-WinEvent -LogName System -MaxEvents 1000 | Group-Object -Property
LevelDisplayName
```

```
Count Name          Group
----- ----          -----
  153 Error         {System.Diagnostics.Eventing.Reader.EventLogRecord,
System.Diag...}
  722 Information   {System.Diagnostics.Eventing.Reader.EventLogRecord,
System.Diag...}
  125 Warning       {System.Diagnostics.Eventing.Reader.EventLogRecord,
System.Diag...}
```

--------- Example 4: Group processes by priority class ---------

```
Get-Process | Group-Object -Property PriorityClass

Count Name         Group
----- ----         -----
   55 Normal       {System.Diagnostics.Process (AdtAgent), System.Diagnosti...
    1              {System.Diagnostics.Process (Idle)}
    3 High         {System.Diagnostics.Process (Newproc), System.Diagnostic...
    2 BelowNormal  {System.Diagnostics.Process (winperf),

Get-Process | Group-Object -Property PriorityClass -NoElement

Count Name
----- ----
   55 Normal
    1
    3 High
    2 BelowNormal
```

-------------- Example 5: Group processes by name --------------

```
Get-Process | Group-Object -Property Name -NoElement | Where-Object {$_.Count
-gt 1}

Count Name
----- ----
2     csrss
5     svchost
2     winlogon
2     wmiprvse
```

---------- Example 6: Group objects in a hash table -----------

```
$A = Get-Command Get-*, Set-* -CommandType cmdlet |
    Group-Object -Property Verb -AsHashTable -AsString
$A

Name     Value
----     -----
Get      {Get-Acl, Get-Alias, Get-AppLockerFileInformation,
Get-AppLockerPolicy...}
Set      {Set-Acl, Set-Alias, Set-AppBackgroundTaskResourcePolicy,
Set-AppLockerPolicy...}

$A.Get

CommandType     Name                            Version    Source
-----------     ----                            -------    ------
Cmdlet          Get-Acl                         3.0.0.0
Microsoft.PowerShell.Security
Cmdlet          Get-Alias                       3.1.0.0
Microsoft.PowerShell.Utility
Cmdlet          Get-AppLockerFileInformation    2.0.0.0    AppLocker
Cmdlet          Get-AppLockerPolicy               2.0.0.0    AppLocker
...
```

Example 10: Group hashtables by their key values with calculated properties

```
@(
    @{ name = 'a' ; weight = 7 }
    @{ name = 'b' ; weight = 1 }
    @{ name = 'c' ; weight = 3 }
    @{ name = 'd' ; weight = 7 }
) | Group-Object -Property { $_.weight } -NoElement

Count Name
----- ----
    2 7
    1 1
    1 3
```

REMARKS

To see the examples, type: "get-help Group-Object -examples".

For more information, type: "get-help Group-Object -detailed".

For technical information, type: "get-help Group-Object -full".

For online help, type: "get-help Group-Object -online"