ForEach-Object

Perform an operation (execute a block of statements) against each item in a collection of input objects, typically passed through the pipeline.

```
Syntax
    ForEach-Object [-process] ScriptBlock[] [-inputObject psobject]
       [-begin scriptblock] [-end scriptblock] [-Parallel] [-ThrottleLimit n] [CommonParameters]

key
   -begin ScriptBlock
       A script block to run before processing any input objects.

   -end ScriptBlock
       A script block to run after processing all input objects.

   -process ScriptBlock[]
       A block of PowerShell script that is applied to each incoming object.

   -inputObject psobject
       Accepts an object that the script block specified in the process parameter will act upon.
       Enter a variable that contains the object(s) or type a command or expression that gets the object(s).

       When the -InputObject parameter is used to submit a collection of items,
       ForEach-Object receives one object that represents the collection.
       Because one collection object cannot be processed, ForEach returns the entire collection unchanged.

       To process multiple items, pipe them to ForEach-Object.

       This parameter is an implementation detail: its purpose is to enable input via the pipeline, and its
       direct use with arrays (collections) does not (yet) provide any useful functionality.

   -Parallel
       Run all script blocks in parallel for each piped input object. PowerShell 7.0+
       Parallel will not always speed up script execution. And can significantly slow down script
       execution if used heedlessly. The overhead for a trivial script can make -parallel much slower.

   -ThrottleLimit
       Limit the number of script blocks running in parallel at a time, default = 5. PowerShell 7.0+
```

Standard Aliases for Foreach-Object: the '**%**' symbol, ForEach

For operations in the pipeline, the ForEach alias will take precedence over the ForEach statement.
For operations not in the pipeline the ForEach statement will take precedence.

For the fastest performance: use the ForEach statement (or method) when the collection of objects is small enough that it can be loaded into memory. (eg an array of 20 string values)

Use the ForEach-Object cmdlet when you want to pass only one object at a time through the pipeline, minimising memory usage. (e.g. a directory containing 10,000 files)

Examples

Retrieve the files (and folders) from the C: drive and display the size of each:

```
PS C:> Get-ChildItem C:\ | ForEach-Object -process { $_.length / 1024 }
```

(The $_ variable holds a reference to the current item being processed.)

Retrieve the 500 most recent events from the system event log and store them in the $events variable:

```
PS C:> $events = Get-Eventlog -logname system -newest 500
```

Then pipe the $events variable into the ForEach-Object cmdlet.

```
$events | ForEach-Object -begin {
   Get-Date
} -process {
   Out-File -filepath event_log.txt -append -inputobject $_.message
} -end {
   Get-Date
}
```

In this example, the -Process parameter uses Out-File to create a text file and stores the .message property of each event.
The -Begin and -End parameters are used to display the date/time before and after the -process command is run.

"The best way to have a good idea is to have a lot of ideas" ~ Linus Pauling

Related PowerShell Cmdlets

ForEach statement.
ForEach (method) - Loop through items in a collection.
Compare-Object Compare the properties of objects.
Group-Object - Group the objects that contain the same value for a common property.
Invoke-Parallel - via the PSParallel module - invoke scriptblocks in parallel runspaces (an alternative to Foreach-Object -parallel).
Measure-Object - Measure aspects of object properties and create objects from those values.
New-Object - Create a new .Net object.
Select-Object - Select objects based on parameters set in the Cmdlet command string.
Sort-Object - Sort the input objects by property value.
Tee-Object - Send input objects to two places.
Where-Object - Filter input from the pipeline allowing operation on only certain objects.
