How to Create a File in PowerShell

lazyadmin.nl/powershell/create-file

September 21, 2023

In PowerShell, there are a couple of ways to create a file. The most common method is to use the `new-item` cmdlet. This cmdlet can not only be used to create files but also folders. But new-item is not always the best option.

Other options to create a file in PowerShell are for example to redirect the results of a cmdlet to a file, or we can use the Set-Content to create a file and store the results.

In this article, we will look at the different options, and I will give you some tips on creating files.

PowerShell Create File

Let’s start with the basics, to create a new file in PowerShell we are going to use the cmdlet `New-Item`. This cmdlet can be used to create files, folders, or symbolic links. We will focus on creating a file in this article.

To create an empty file we will at least specify the name of the file and set the `-ItemType` property to file. If you don’t specify the path where you want to create the file, then the file will be created in the current directory.

```
New-Item -Name "test01.txt" -ItemType File
# Create a file in c:\temp
New-Item -Path "c:\temp" -Name "test01.txt" -ItemType File
```

The path parameter also accepts wildcards. Now you might think, why would I need a wildcard in the path, but this allows you for example to create a new file in every subfolder of the path:

```
New-Item -Path C:\Temp\* -Name temp.txt -ItemType File
# Result
C:\Temp\One\temp.txt
C:\Temp\Three\temp.txt
C:\Temp\Two\temp.txt
```

The new-item cmdlet won’t create a file if it already exists. In some cases you might just want to overwrite the file or make sure that your script continues without throwing any errors. If you want to overwrite a file if it already exists, then you can use the -force parameter.

This will overwrite the file if it exists, which also means that any content in the file will be lost:

```
New-Item -Path "c:\temp" -Name "test01.txt" -ItemType File -Force
```

Another option is to check if the file exists, we can do this with the Test-Path cmdlet:

```
$path = "c:\temp\test1.txt"
# Create the folder is the path doesn't exist
If (-not(test-path -Path $path)) {
New-Item -Path $path -ItemType File
}
```

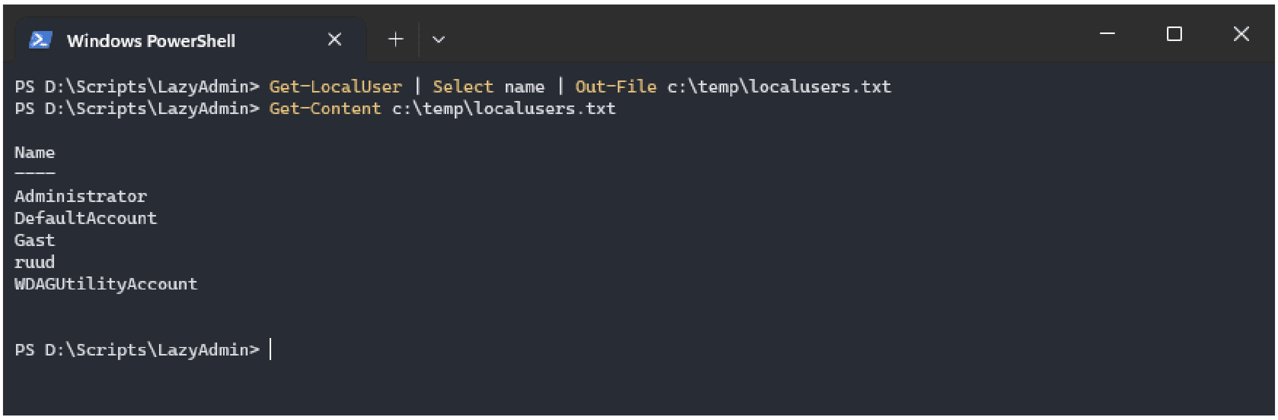
As you can see in the example above, we actually don't have to specify the filename, we can also include it in the file path. The New-Item cmdlet will then still create the required file.

Create a File with Content

As mentioned in the beginning, there are other methods to create a file in PowerShell. These methods can be used when you not only want to create the file but also want to add content to it. For example, when you retrieve data with a cmdlet and want to store it in a text file, then we can use the `Out-File` cmdlet or use a redirect operator

The Out-File cmdlet will redirect the results of the cmdlet to the specified file. If the file doesn't exist, then it will create it. Let's say we want to get all local users from our computer and want to store it in a text file:

```
Get-LocalUser | Select name | Out-File c:\temp\localusers.txt
# Or with the redirect operator
Get-LocalUser | Select name > c:\temp\localusers.txt
```



```
PS D:\Scripts\LazyAdmin> Get-LocalUser | Select name | Out-File c:\temp\localusers.txt
PS D:\Scripts\LazyAdmin> Get-Content c:\temp\localusers.txt

Name
----
Administrator
DefaultAccount
Gast
ruud
WDAGUtilityAccount


PS D:\Scripts\LazyAdmin> |
```

Read more about the `Out-File` cmdlet, and how to append content for example, in this article.

We can also create a file and save the content with the `New-Item` cmdlet or the `Set-Content` cmdlet. Both cmdlets work pretty much the same, only the Set-Content cmdlet will replace the contents when the file already exists, whereas `New-Item` will throw an error when the file already exists.

```
# Create a file and set the content
Set-Content -Path c:\temp\newFile.txt -Value "New file with Content"
# Or use New-Item to create the file with content
```

```
New-Item -Path c:\temp\ -name "newFile.txt" -Value "New file with Content"
```

Creating Multiple Files

To create multiple files we have two options, we can either specify each file inside the **path** parameter or use an ForEach loop. If you only need to create two or three files then you could just specify them in the path parameter:

```
New-Item -Path "c:\temp\test01.txt","c:\temp\test02.txt" -ItemType File
```

But a more readable option is to use a small ForEach loop and an array with the files that you want to create. This way it is easier to add or change files in the array and also add some logic to it when needed.

```
# Specify the directory where you want to create the files
$targetDirectory = "C:\temp\"
# Create an array of file names
$fileNames = @("File1.txt", "File2.txt", "File3.txt")
# Loop through each file name and create a file with content
foreach ($fileName in $fileNames) {
New-Item -Path $targetDirectory -Name $filename -ItemType file
$filePath = Join-Path -Path $targetDirectory -ChildPath $fileName
Write-Host "Created file: $filePath"
}
```

Another option, when you don’t want to specify the file names, is to iterate over an index and create some text files using the index number:

```
1..10 | ForEach-Object {
New-Item -Path "C:\temp\" -Name "$_.txt" -ItemType file
}
```

Creating Special Files

Until now we have only created TXT files. But with PowerShell, we are not limited to TXT files alone. We can also create CSV, JSON, HTML, or XML files for example. To do this we will need to make sure that the content of the file is converted to the correct format.

Basically, you can create any file type with the New-Item cmdlet. If you would for example set the file type to `.pdf`, then you will get a “PDF” file in the given directory. The only problem is that you can’t open it, because the content of the file is not in the correct format.

This means that if you want to create a CSV file for example, that you will need to use the Export-CSV cmdlet. Or for an XML file, you will need to use the `Export-CliXml` cmdlet:

```
$xmlData | Export-CliXml -Path "c:\temp\test.xml" -Encoding UTF8
```

Wrapping Up

Creating files in PowerShell is pretty easy with the new-item cmdlet. Make sure that the contents of your file match the file type, by using the correct cmdlets to store the data inside the file.

I hope you found this article helpful, if you have any questions, just drop a comment below.