How to Wait for a Command in PowerShell

lazyadmin.nl/powershell/wait-command

October 4, 2024

When you run a command or start a process from PowerShell, you often want to wait for the command or process to finish, before the script continues. The easiest option for this is to use the -Wait parameter from the Start-Process cmdlet, but there are other options as well.

In this article, we will look at how to use the -Wait parameter, use the Wait-Process cmdlet and I will show you some other options as well.

Wait in PowerShell

We can use PowerShell for all kinds of tasks, including starting a program or running another command. But the “problem” with PowerShell is, that it by default won’t wait for the program or command to finish.

It will just try to start the program or run the command, and if that is successful, it will continue to the next part of your script. Now in some situations, you want PowerShell to wait before it continues. To do this, we have a couple of options:

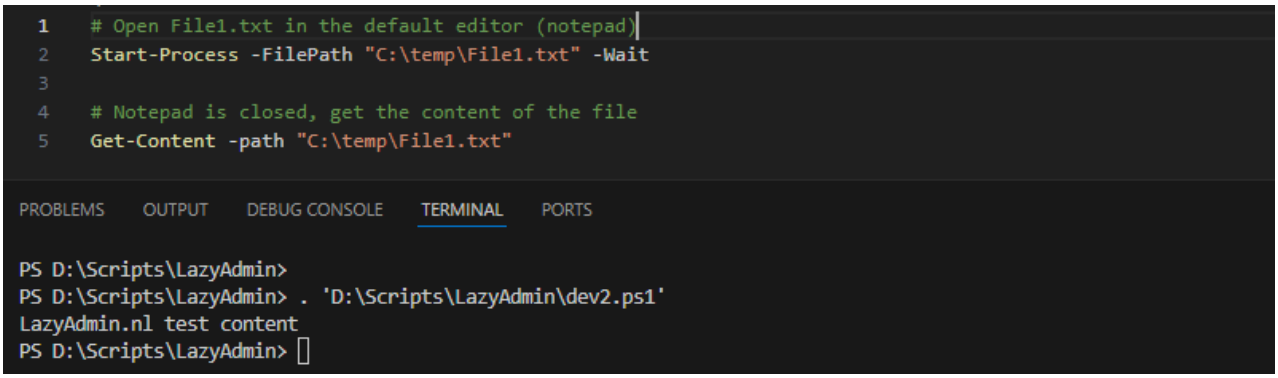
- **-Wait parameter** for the Start-Process cmdlet
- **Wait-Process**
- **Start-Sleep**

Start-Process with -Wait

When you want to run a program or command from PowerShell, you can use the `Start-Process` cmdlet. Let’s take a look at an example, we are going to open a text file in Notepad, and wait for the user to close the file, before we continue with the script:

```
# Open File1.txt in the default editor (notepad)
Start-Process -FilePath "C:\temp\File1.txt" -Wait
# Notepad is closed, get the content of the file
Get-Content -path "C:\temp\File1.txt"
```

If you run the above code in PowerShell, you will see that Notepad will open, allowing you to write content in the file, and after you close it, the contents of the file will be displayed in the console.



Wait-Process

Another option is to use the `Wait-Process` cmdlet. This cmdlet waits for one or more processes to be closed before it allows PowerShell to continue. Important to note, that the Wait-Process cmdlet doesn't work on Linux or macOS.

To check if a process is closed, we first need to know the process name or process ID. Another option is to use a process object, which we for example can get with the `Get-Process` or `Start-Process` cmdlet.

If you know the process name and the name is unique, then you can simply use the `-name` parameter to get and wait for the process to finish:

Wait-Process Name "Notepad"
Another option is to first look up the process, using the `Get-Process` cmdlet. This cmdlet will list all running processes, but we can also filter the processes on name using the `-name` parameter. Good to know is that we can use wildcards in the name:

$proc = Get-Process -Name "Notepad"
Wait-Proces -Id $proc.id
We can also combine Start-Process and Wait-Process. To do this, we will need to use the `-PassThru` parameter for the Start-Process cmdlet so it returns the process object.

This allows us to do comes while the process is running, but check later in the script if the process is finished before we continue with the last part:

Start the process
$process = Start-Process -FilePath "notepad.exe" -ArgumentList "C:\temp\File1.txt" -PassThru
Do something in between
Write-Output "Process $($process.Id) is running."
Wait untill process is finished
Wait-Process -Id $process.Id
Write-Output "Process has finished."

Start-Sleep

The last method that you could use to wait if a process is finished in PowerShell is the Start-Sleep method. This cmdlet isn't able to check if a process is running or not but only lets PowerShell wait for an x amount of time.

This can still be pretty useful if you for example know that a process always needs a couple of seconds to get up and running or to finish.

The `Start-Sleep` cmdlet works by specifying how long, in seconds or milliseconds, PowerShell should wait.

For example, to let PowerShell Wait for 5 seconds, we can use:

```
# Wait for 5 seconds before continuing
Start-Sleep -Seconds 5
```

Wait for Input

Another method that I want to show you as well is to wait for input in PowerShell. Sometimes it's just easier to let PowerShell wait for a keypress before it continues. This can be any key input, or a specific key if you want. We have two options to wait for input, we can use the `ReadKey` method from the `System.Console .Net` class or we can use the `Read-Host method`.

The ReadKey method below won't return any output in the console (except for the write-host of course), but it doesn't work in PowerShell ISE (but I recommend using Visual Studio Code anyway..)

```
Write-Host "Press any key to continue..."
$Host.UI.RawUI.ReadKey("NoEcho,IncludeKeyDown") | Out-Null
```

With Read-Host, however, we can specify which key the user must press. This give a bit more control over when to continue the script. In the example below, we combine a While loop with the Read-Host cmdlet, so it will keep asking to press the `Y` key before PowerShell continues.

```
while ((Read-Host "Press 'Y' to continue") -ne 'Y') {}
```

Wrapping Up

When start the process or command from PowerShell, it's best to use the Start-Process cmdlet combined with the -Wait parameter. If the process is already running, then the Wait-Process cmdlet is the best option to use.

Hope you liked this article, if you have any questions, just drop a comment below.

Did you **Liked** this **Article**?
Get the latest articles like this **in your mailbox**
or share this article

Не удается связаться с сервисом reCAPTCHA. Проверьте подключение к Интернету и перезагрузите страницу.

I hate spam to, so you can unsubscribe at any time.