Powershell Sleep – Pause your Script

lazyadmin.nl/powershell/start-sleep

April 15, 2021

Sometimes you will need to add suspend or pause your PowerShell script for a couple of seconds. For example when you are using a do-while loop to check if a server is back online. To do this we can use the PowerShell **Start-Sleep cmdlet**. With this command, we can let the script sleep for a couple of seconds.

In this article, I am going to give you a couple of examples of how you can use the PowerShell Sleep cmd.

Using the PowerShell Start Sleep cmdlet

The start-sleep cmdlet in PowerShell is actually pretty straightforward to use. We only need to define how long we want to suspend the script. We can define the sleep time in **seconds (default)** or **milliseconds**.

To let your PowerShell script pause for 2,5 seconds you can do the following:

```
# Suspend the script for 2.5 seconds
Start-Sleep -Seconds 2.5
```

The parameter `-Seconds` isn't really needed, by default seconds are used. So you can also only specify the number of seconds `Start-Sleep 5`. But for readability is preferred to use the `-s` or `-seconds` parameter.

```
# Pause for 5 seconds per loop
Do {
# Do stuff
# Sleep 5 seconds
Start-Sleep -s 5
}
while ($condition -eq $true)
```

The only other parameter besides seconds is milliseconds. This allows you to let your script wait for only a short amount of time.

```
# Pause for 0.1 second per loop
Do {
# Do stuff
# Sleep 100 Milliseconds
Start-Sleep -Milliseconds 100
}
while ($condition -eq $true)
```

The most common place to use the start-sleep cmdlet is in a PowerShell loop. But you can use the sleep cmdlet anywhere in your script. The only problem with that is that you are probably guessing how long you should wait.

> Tip
>
> *You can break-out of a sleep with the key combination Ctrl + C. This will resume your PowerShell session or script.*

Sleep Until a Specific Time

Running a PowerShell script for a long period of time isn't always the best option, but if you need to pause your PowerShell script until a specific time, then we can calculate the number of seconds to sleep using two date objects.

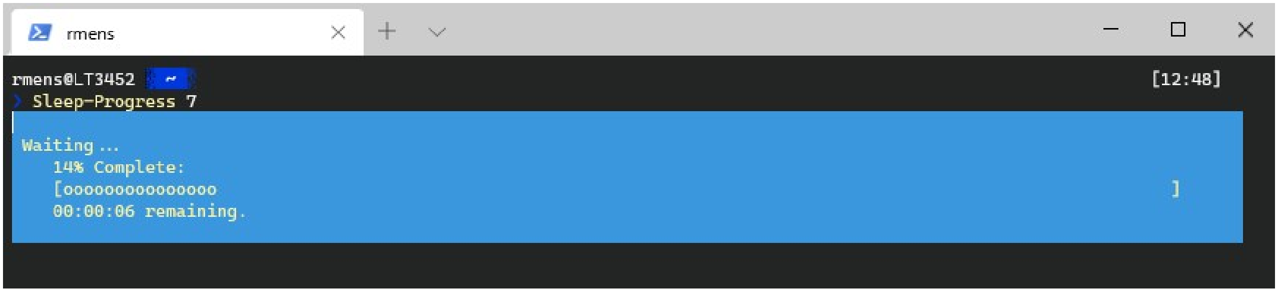
```
Start-Sleep ((Get-Date "06:00pm") - (Get-Date)).TotalSeconds;
```

If you are using this method, then it might be a good idea to add a warning or message in your script as well, so you know how long it will sleep:

```
$sleepUntil = "06:00pm"
$secondsToSleep = ((Get-Date $sleepUntil) - (Get-Date)).TotalSeconds
Write-Host "Pause script for $secondsToSleep seconcds untill $sleepUntil"
Start-Sleep $secondsToSleep
```

PowerShell Sleep with Progressbar

When you pause a PowerShell script with the `start-sleep` cmdlet then you won't see anything happening in the console. To give the user some visual feedback you could add a progress bar to your sleep command.



```
rmens@LT3452  ~                                                                    [12:48]
❯ Sleep-Progress 7

Waiting ...
   14% Complete:
   [ooooooooooooooo                                                                 ]
   00:00:06 remaining.
```

The only problem with this is that you need to write a lot of code to create the progress bar.

What you can do is create a function, that will suspend your script for the desired time and show the progress bar. We are using the `Write-Progress` cmdlet for this and calculating the percentage and time remaining in a do-while loop.

You can then simply call the function Sleep-Progress to suspend your script and show the progress bar:

```
Function Sleep-Progress($seconds) {
$s = 0;
Do {
$p = [math]::Round(100 - (($seconds - $s) / $seconds * 100));
Write-Progress -Activity "Waiting..." -Status "$p% Complete:" -SecondsRemaining
($seconds - $s) -PercentComplete $p;
[System.Threading.Thread]::Sleep(1000)
$s++;
}
While($s -lt $seconds);
}
Sleep-Progress 7
```

Wrapping Up

When you need to suspend or pause a PowerShell script, then Start-Sleep is the way to go. I hope you found this article useful, if you have any questions, then just drop a comment below.

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

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