

How to use Powershell Write Host

The PowerShell Write-Host cmdlet is used to output text to the console for the user. It's commonly used to show the progress of a script, output results, or show instructions for the user.

In this article, we are going to take a look at how to use the Write-Host cmdlet, the different options that it comes with, and I will show you a couple of examples.

PowerShell Write-Host CmdLet

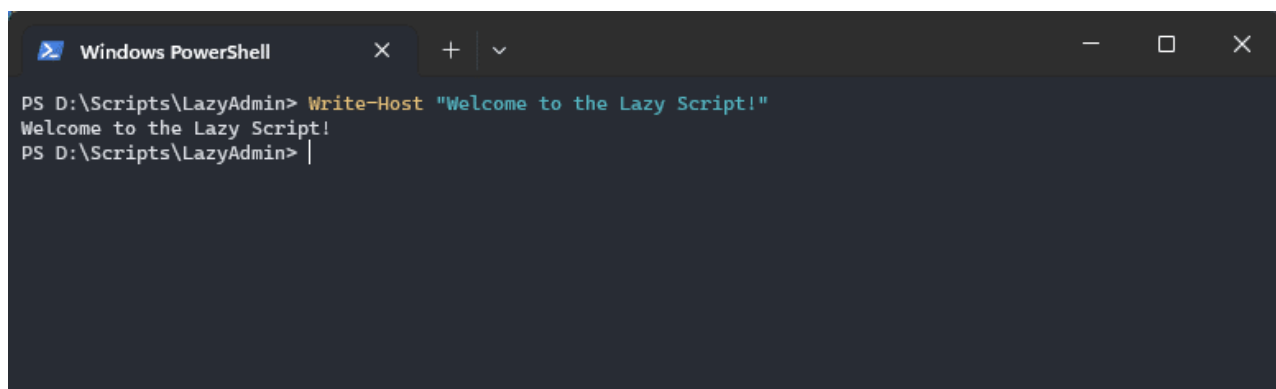
The `Write-Host` cmdlet is actually a wrapper for the older `Write-Information` cmdlet. Until PowerShell 5.0 we had to use the `Write-Information` or `Write-Output` cmdlet to output information to the console. The old cmdlets didn't come with a lot of features, whereas `write-host` gives us a couple of styling options that we can use:

Parameter	Description
-ForegroundColor	Color of the text
-BackgroundColor	Background color of the text
-NoNewline	Won't add a newline after the output
-Separator	Adds a separator between the output of each item in an objects or array

Write-Host CmdLet Parameters

By default, the `Write-Host` cmdlet will output a string on a new line in the console, using the default color of the console (most of the time white):

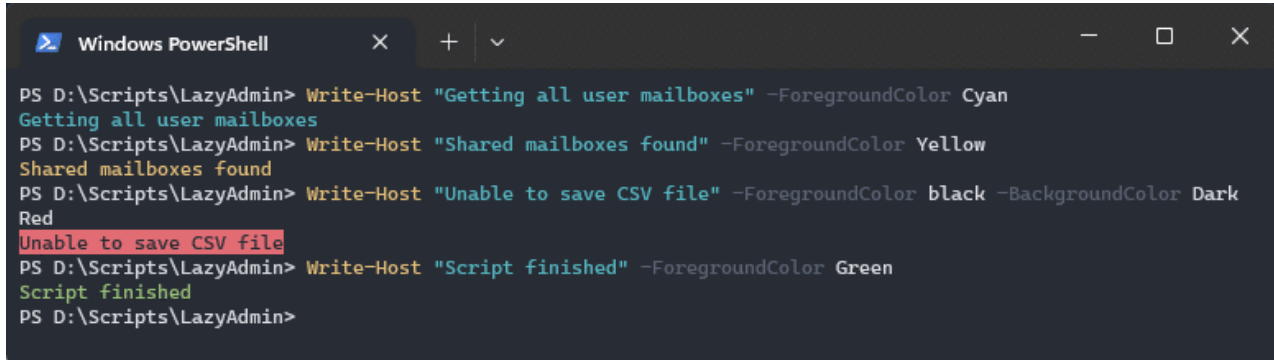
Write-Host "Welcome to the Lazy Script!"



```
Windows PowerShell
PS D:\Scripts\LazyAdmin> Write-Host "Welcome to the Lazy Script!"
Welcome to the Lazy Script!
PS D:\Scripts\LazyAdmin> |
```

Using Colors in Write-Host

Colors are a great way to attract the user's attention or tell something about the status non-verbally. For example, yellow text often indicates a warning whereas red text is mostly likely an error. In PowerShell we can assign a color for the text, using the `-foregroundColor` parameter and a background color for the text with `-backgroundColor`.



```
Windows PowerShell
PS D:\Scripts\LazyAdmin> Write-Host "Getting all user mailboxes" -ForegroundColor Cyan
Getting all user mailboxes
PS D:\Scripts\LazyAdmin> Write-Host "Shared mailboxes found" -ForegroundColor Yellow
Shared mailboxes found
PS D:\Scripts\LazyAdmin> Write-Host "Unable to save CSV file" -ForegroundColor black -BackgroundColor DarkRed
Unable to save CSV file
PS D:\Scripts\LazyAdmin> Write-Host "Script finished" -ForegroundColor Green
Script finished
PS D:\Scripts\LazyAdmin>
```

We can use the following colors for both parameters:

- Black
- DarkBlue
- DarkGreen
- DarkCyan
- DarkRed
- DarkMagenta
- DarkYellow
- Gray
- DarkGray
- Blue
- Green
- Cyan
- Red
- Magenta
- Yellow
- White

Formatting the Output

By default, each string that is outputted by Write-Host is displayed on a new line. In most cases this is fine, but sometimes you don't want a new line, or maybe an extra new line between the outputs.

Let's first take a look at how to append the results behind the previous Write-Host output. To do this we can use the `-NoNewline` parameter. When added, the next output will append to the existing line:

```
Write-Host "Number of mailboxes: " -NoNewline
# Get all mailboxes function
```

```
$mailboxes = Get-AllMailboxes
```

```
Write-Host $mailboxes
```

```
# Result:
```

```
Number of mailboxes: 5
```

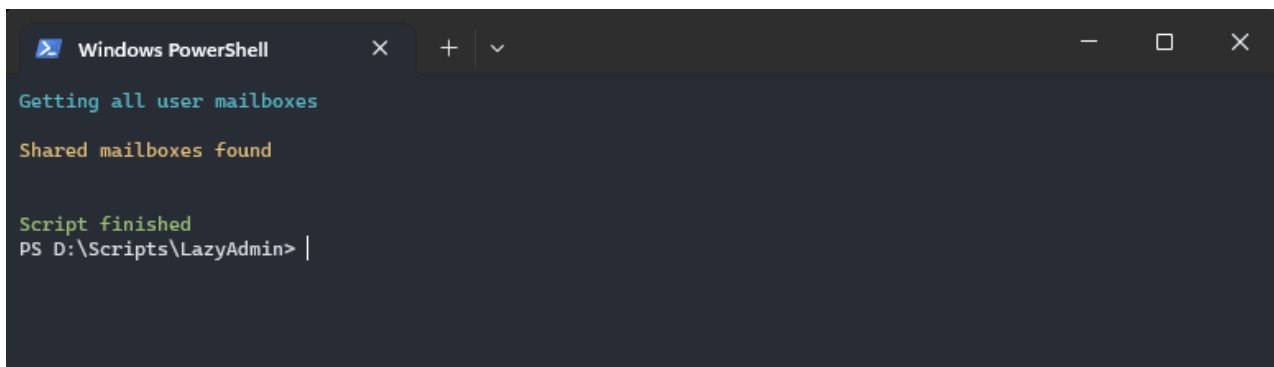
As you can see, the value `$mailboxes` is appended behind the line "Number of mailboxes".

If you want to create an extra new blank line between the outputs, then we can use the ``n` character. This will add an extra new line behind the output of the `write-host` cmdlet:

```
Write-Host "Getting all user mailboxes `n" -ForegroundColor Cyan
```

```
Write-Host "Shared mailboxes found `n `n" -ForegroundColor Yellow
```

```
Write-Host "Script finished" -ForegroundColor Green
```



```
Windows PowerShell
Getting all user mailboxes
Shared mailboxes found
Script finished
PS D:\Scripts\LazyAdmin> |
```

Using Variables inside Write Host

Good to know is that we can use variables inside the output of the `write-host` cmdlet. A simple string variable can be included in the string of `write-host` when using the double-quotes `" "`. Keep in mind that you can't use variables inside strings wrapped with single-quotes `' '`. The difference here is that PowerShell only processes strings that are wrapped in double quotes.

```
$user = "megan@lazyadmin.nl"
```

```
Write-Host "Getting mailbox of $user"
```

```
# Result
```

```
Getting mailbox of megan@lazyadmin.nl
```

But when working with objects, you might have noticed that you can't output the items of an object. For example, this **won't** work:

```
$user = [PSCustomObject]@{
```

```
  DisplayName = "Megan"
```

```
  Email = "megan@lazyadmin.nl"
```

```
}
```

```
Write-Host "Getting mailbox of $user.displayname"
```

To show the display name of the user object, we will need to wrap it in parentheses, like this:

```
$user = [PSCustomObject]@{
```

```
DisplayName = "Megan"
Email = "megan@lazyadmin.nl"
}
Write-Host "Getting mailbox of $($user.DisplayName)"
# Result
Getting mailbox of Megan
```

Using a Separator

When you want to output the results of an array to the console, then the results will be displayed on a single row by default. For example, I have a CSV file with some fruits. When we import the CSV file and output the result then it will be formatted like this:

```
$example = import-csv -path C:\temp\example.csv -Header Fruit
Write-Host $example.fruit
# Result
```

```
Apple Banana Pear Kiwi Raspberry Melon
```

To show each fruit on its own line or separate each fruit with a character, we can use the **-Separator** parameter:

```
# Show results on a new line
$example = import-csv -path C:\temp\example.csv -Header Fruit
Write-Host $example.fruit -Separator "`n"
# Result
```

```
Apple
Banana
Pear
Kiwi
Raspberry
Melon
```

```
# Separate results with a space and dash
$example = import-csv -path C:\temp\example.csv -Header Fruit
Write-Host $example.fruit -Separator " - "
# Result
Apple - Banana - Pear - Kiwi - Raspberry - Melon
```

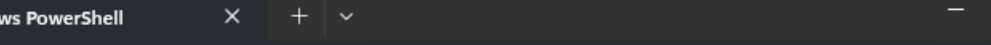
Write Host Formatting Examples

With some creativity, you can create different banners, or alerts in your scripts that really stand out. Below I gather a couple of examples as inspirations for you. If you have other good examples, please share them in the comments below!

Adding a new line above and below a warning message, adding some space around the message, and using a background color can really make a message stand out:

```
Write-Host "`n"
```

```
Write-Host " RUNNING IN TEST MODE " -BackgroundColor Yellow -ForegroundColor
Black
Write-Host "`n"
```



The screenshot shows a Windows PowerShell terminal window with a dark background. The title bar at the top reads "Windows PowerShell" and includes standard window controls (close, maximize, and a dropdown arrow). The command prompt shows the user is in the directory "D:\Scripts\LazyAdmin" and has executed the command ".\dev.ps1". Below the command, a yellow banner displays the text "RUNNING IN TEST MODE". The prompt returns to "PS D:\Scripts\LazyAdmin>".

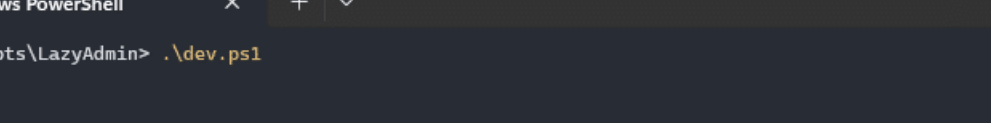
```
PS D:\Scripts\LazyAdmin> .\dev.ps1
```

RUNNING IN TEST MODE

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

Script banners are always a bit of a puzzle to get right. In the code example below it doesn't look aligned, but when using it in a script you will see a perfect banner.

```
Write-Host "`n"
Write-Host "-----" -ForegroundColor Cyan
Write-Host " | " -ForegroundColor Cyan
Write-Host " | Create new AD User | " -ForegroundColor Cyan
Write-Host " | Version 1.7 | " -ForegroundColor Cyan
Write-Host " | " -ForegroundColor Cyan
Write-Host " | Author R. Mens - LazyAdmin.nl | " -ForegroundColor Cyan
Write-Host " | " -ForegroundColor Cyan
Write-Host "-----" -ForegroundColor Cyan
Write-Host "`n"
```



The screenshot shows a Windows PowerShell terminal window with a dark background. The title bar at the top reads "Windows PowerShell" and includes standard window controls (close, maximize, and a dropdown arrow). The command prompt shows the user is in the directory "D:\Scripts\LazyAdmin" and has executed the command ".\dev.ps1". The output of the script is displayed in a light blue monospaced font, enclosed within a dashed rectangular border. The output text reads: "Create new AD User", "Version 1.7", and "Author R. Mens - LazyAdmin.nl". The prompt "PS D:\Scripts\LazyAdmin>" is visible at the bottom of the terminal.

```
Windows PowerShell
PS D:\Scripts\LazyAdmin> .\dev.ps1

Create new AD User
Version 1.7

Author R. Mens - LazyAdmin.nl

PS D:\Scripts\LazyAdmin>
```

We can also do something fun with the foreground colors and some ASCII art. The code below divides the position of each character with 6. Depending on the result, it will assign a color to the character. By using the `-NoNewline` parameter we can append the results:

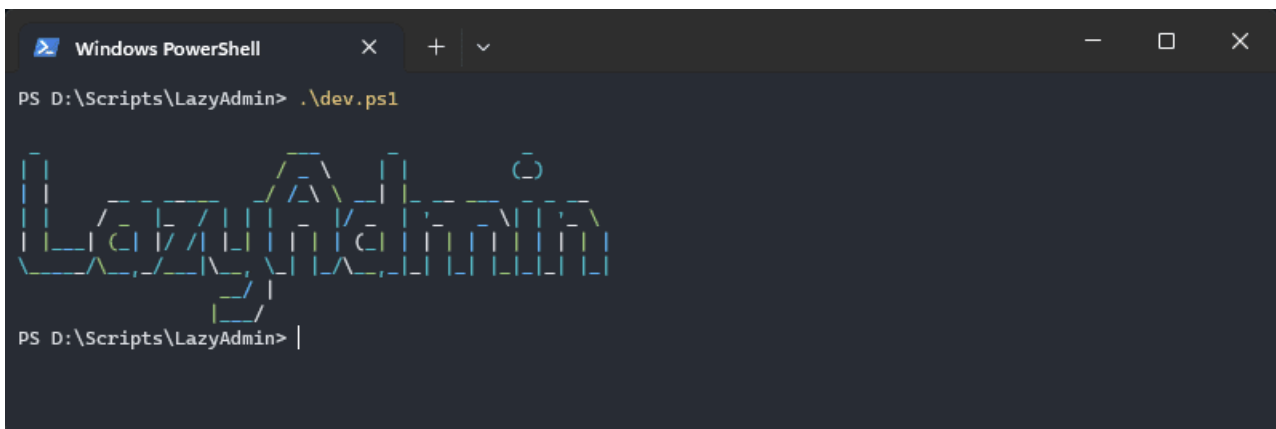
```
$text = "
```

```
|_|/_\||(|)
|_|_ _ _ _ _ /_/\\ |_|_ _ _ _ _
|_|/_ _ |/_|||_|/_|'_ _ \|'_ _ \
|_|_ _ |( |/_/||| |||( | ||||||||
```

```

\____^_,/_\_,\|\^_,|| || || || ||
_/|
|_/ "
for ($i=0; $i -lt $text.length; $i++) {
switch ($i % 6) {
0 { $c = "white" }
2 { $c = "green" }
4 { $c = "blue" }
default { $c = "cyan" }
}
write-host $text[$i] -NoNewline -ForegroundColor $c
}

```



Wrapping Up

The Write-Host cmdlet is a great way to keep the user of your script informed about the progress of your scripts. It can give instructions to the user, show the results, or simply inform the user when the script is completed or runs into an error.

Make good use of the foreground colors, because they really help with quickly identifying the meaning of a message.

I hope you like 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

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