

# How to Concatenate a String in PowerShell

---

 [lazyadmin.nl/powershell/concatenate-string](https://lazyadmin.nl/powershell/concatenate-string)

December 8, 2022

When outputting data from PowerShell you often need to concatenate two or more strings or variables. Joining strings in PowerShell is primarily done using the `+` operator. But there are other ways to concatenate strings as well.

The problem with joining strings or variables in PowerShell is often finding the correct output method. You might get need extra space between the two strings, or you get the `+` sign in the output, which you don't want of course.

In this article, we will look at the different methods to join multiple strings in PowerShell.

## Powershell Concatenate String

---

The basic method to concatenate a string in PowerShell is to use the `+` operator. When using the `+` operator, the two strings will simply be joined together. Let's take the following two strings to start with:

```
$string1 = "The quick brown fox"
```

```
$string2 = "jumps over the lazy dog"
```

Now if we want to output these strings together in the PowerShell console, people often tend to do the following:

```
Write-Host $string1 + $string2
```

```
# Result
```

```
The quick brown fox + jumps over the lazy dog
```

But if you look at the result, you will see that the `+` sign is also in the output, which we don't want. The correct way to output both strings to the console is by simply placing them after each other.

```
Write-Host $string1 $string2
```

```
# Result
```

```
The quick brown fox jumps over the lazy dog
```

But this is not concatenating strings. To actually join the strings together we can store the result first in another variable using the `+` operator:

```
$string1 = "The quick brown fox"
```

```
$string2 = "jumps over the lazy dog"
```

```
$result = $string1 + $string2
```

```
Write-Host $result
```

```
# Result
```

```
The quick brown foxjumps over the lazy dog
```

The only issue here is that you miss the space between the two strings. So to solve that you could concatenate also a space between the two strings:

```
$result = $string1 + " " + $string2
```

## Using the Join Operator

---

Another method to join strings in PowerShell is to use the Join Operator. The `join` operators allow you to concatenate two or more strings with a specified separator. Taking our two example strings, we can do the following:

```
$result = $string1,$string2 -join " "
```

```
# Result
```

```
The quick brown fox jumps over the lazy dog
```

Another option is to use the `.Net join` method. The principal and results are the same as the join operator, only the writing style is a bit different:

```
[string]::Join(' ', $string1, $string2)
```

## Using the Format Operator

---

When you need to insert multiple variables into a string, then using the string format operator might be a good option to use. The advantage of the format operator `-f` is that it is easier to read and format with longer strings or multiple variables.

The format operator uses placeholders to determine where you want to insert a string or variable. The placeholders are numbered upwards from 0 and placed between curly brackets.

Take the example below, we have a `PSCustomObject` with a couple of strings. We can concatenate these strings in PowerShell into another string using the string format operator.

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

```
  Name = 'Lazy Dog'
```

```
  Action = 'jump over'
```

```
  Item = "3 fences"
```

```
}
```

```
write-host ("can the {0} really {1} more than {2}" -f $obj.name, $obj.action, $obj.item)
```

```
# Result
```

```
can the Lazy Dog really jump over more than 3 fences
```

You don't need to use variables for the right side of the `-f` operator. You can also insert strings or integers directly:

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

```
  Name = 'Lazy Dog'
```

```
  Action = 'jump over'
```

```
  Item = "fences"
```

```

}
write-host ("can the {0} really {1} {2} {3} {4}" -f $obj.name, $obj.action, 'less than', 5,
$obj.item)
# Result
can the Lazy Dog really jump over less than 5 fences
If you have an array with strings that you want to concatenate in PowerShell, then you
can also use the format operator, without the need of specifying each individual value for
the placeholder.

$values = @(
"Lazy Dog"
"Jumps"
"fences"
)
'Did you know that the {0} {1} 5 {2}?' -f $values
# Result
Did you know that the Lazy Dog Jumps 5 fences?

```

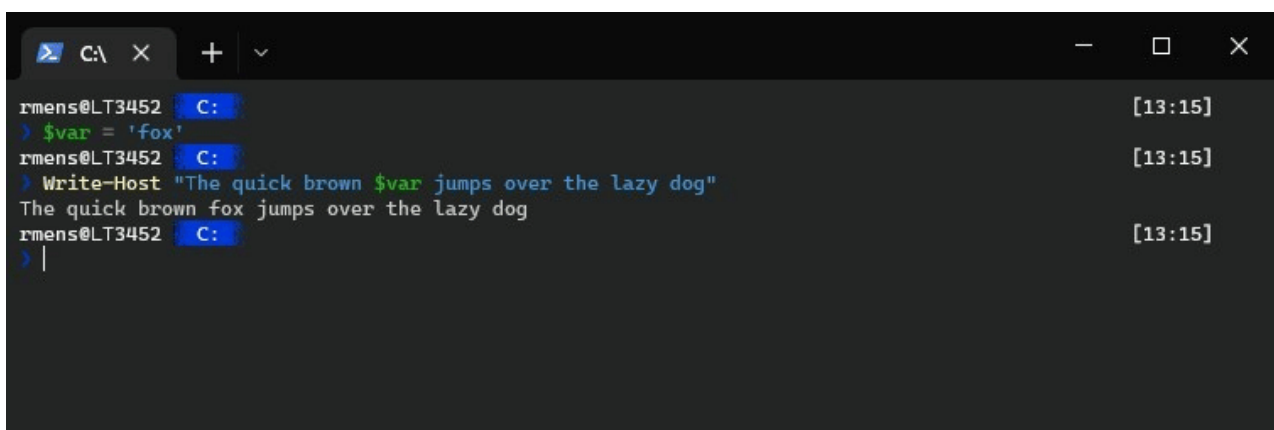
## Concatenate String with Variable

Concatenating strings with variables, and especially objects or hashtables, is always a bit challenging. If you have a single variable that you would like to insert into a string, then you can simply place the string between double quotes “, and insert the variable directly:

```

$var = 'fox'
Write-Host "The quick brown $var jumps over the lazy dog"
# Result
The quick brown fox jumps over the lazy dog

```



The screenshot shows a PowerShell terminal window with the following commands and output:

```

rmens@LT3452 C:
> $var = 'fox'
rmens@LT3452 C:
> Write-Host "The quick brown $var jumps over the lazy dog"
The quick brown fox jumps over the lazy dog
rmens@LT3452 C:
> |

```

powershell concatenate string

But when you do this with the properties of an object, then it won't work. To solve this you will need to use variable substitution, where you place the variable inside `$()`.

```

$obj = [PSCustomObject]@{
Name = 'Lazy Dog'
Action = 'jump over'
}

```

```
Item = "fences"
}
Write-Host "The quick brown fox jumps over the $($obj.name)"
# Result
The quick brown fox jumps over the Lazy Dog
```

## Using the Concat Method

---

In PowerShell, we can also use the .Net concat method to concatenate strings. The advantage of the Join operator or method is that you don't need to specify a separator, but that also makes the use case a bit limited.

```
$values = @(
"Lazy Dog"
"Jumps"
"fences"
)
[string]::Concat('Lazy Dog','Jumps','fences')
# Or when you have an array with values:
[string]::Concat($values)
# Result for both methods:
Lazy DogJumpsfences
```

## Wrapping Up

---

There are a lot of ways in PowerShell to concatenate a string, even more than I have described here. Between these methods, there is little difference when it comes to performance for use cases. So the method to use really depends on the situation and writing style.

When you only need to concatenate a couple of strings, then using the + operator is often the preferred way. When you need to join multiple strings or values, then the format operator (placeholder method) is often easier to read.

Also keep in mind that when you want to output object properties, you will need to wrap them into parentheses `$()` and use double-quotes to wrap the string.

I hope this article helped you with joining strings in PowerShell. 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.