Episode 4

PHP Strings

Introduction

A string is simply a series of characters. Strings are widely used in computer programming, and it is important that as a developer you learn how to create and manipulate them. Because they are important, there are several functions that PHP provides to make our jobs easier when working with strings. In this episode, we will learn about variable interpolation, how to escape special characters and some of the most common functions used on strings.

Goals

In this section of the course your goals are:

- ☐ To learn how to initialize a variable with a string value.
- ☐ To learn the difference between single quotes and double quotes when using strings.
- ☐ To learn about variable interpolation or expansion.
- ☐ To learn about escape sequences for special characters.
- ☐ To learn how to concatenate a string.
- ☐ To learn about the most common functions PHP has to handle strings.

Initializing a String

There are two ways to initialize a string:

- Within single quotes:

```
$name = 'Brooke';
```

- And within double quotes:

```
$title = "Princess";
```

It is as simple as that. Surround the characters that you want on your string variable within single quotes or within double quotes, and you got yourself a string.

Single or Double Quotes. Variable Expansion

Let's now see how we can output these two variables to the screen:

```
echo '<h2>', $name, ' is the ', $title, ' of an enchanted fairy land.</h2>';
```

Let's take a closer look at the syntax we used to output this string. Each of the comma-separated parts is called an argument. Notice that the names of the variables themselves are not surrounded by quotes. The echo statement can take several arguments, and we get the output we intended.

```
Brooke is the Princess of an enchanted fairy land.
```

But what if we wanted to pass just a single argument. Let's see the output of that using single quotes:

```
echo '<h2> $name is the $title of an enchanted fairy land.</h2>';
```

This time we don't get the output we intended:

```
$name is the $title of an enchanted fairy land.
```

We can solve this problem very easily. All we have to do is use double quotes to surround the whole string:

```
echo "<h2> $name is the $title of an enchanted fairy land.</h2>";
```

And once again we get the desired output:

```
Brooke is the Princess of an enchanted fairy land.
```

This was an example of variable interpolation or expansion. When you include a variable name inside single quotes, the output will be the **name** of the variable. To output, the **value** of the variable always use double quotes.

What happens when you have a quote inside a quote. Suppose you have:

```
Mark Twain once said, "The report of my death was an exaggeration".
```

To be able to output this correctly, since we are using double quotes for the inside quote, use single quotes to surround the whole sentence. Conversely, if we were using single quotes for the inside quote, then we would use double quotes to surround the whole sentence.

```
echo 'Mark Twain once said "The report of my death was an exaggeration".<br>';
echo "Mark Twain once said 'The report of my death was an exaggeration'.<br>";
```

Special Characters

Certain characters require a special way of including them in a string for them to be shown correctly. These include the backslash (\ ), linefeed (\n), horizontal tab (\t), vertical tab (\v), carriage return (\r), the dollar sign ($), double quotes (") and single quotes (’). Notice that we preceded each character by a backslash. Of these, only the backslash and single quotes will work **inside single quotes**. Let’s look at the following examples:

```
echo 'Arnold said \'I\'ll be back\'.';
echo 'The file path is C:\\myfile.php';
```

The output will be:

```
Arnold said 'I'll be back'.
The file path is C:\myfile.php
```

All other escape sequences for special characters will only work inside **double quotes**. For example, the following code:

```
echo 'The book cost \$20.00';
echo "The book cost \$20.00";
```

Will output:

```
The book cost \$20.00
The book cost $20.00
```

The escape sequence `\$` only works inside double quotes.

String Concatenation

Suppose we have the following string variables:

```
$subject = "I";
$verb = "like";
$object = "PHP";
```

How do we put these three variables together in a single string. The answer is concatenation, and to do that we use the dot (.) operator:

```
$sentence = $subject . ' ' . $verb . ' ' . $object;
```

And now we have a single variable. Notice that we included a space in single quotes between the words; otherwise, the result would have been “IlikePHP”.

Common String Functions

These are some of the more common functions used to manipulate strings.

Function	Purpose
strlen( )	Find the length of a string
str_word_count( )	Find the number of words in a string
strrev( )	Reverse the characters in a string
strpos( )	Find the position of a substring
str_repeat( )	Repeat a string
str_replace( )	Replace a substring within a string

Here are some examples:

```php
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>String Functions</title>
    <style> body{margin: 30px;} </style>
</head>
<body>
    <h3>
    <?php
    echo "String: 'I really like this PHP course.'", '<br>';

    // Find the length of a string
    echo "Length: ", strlen('I really like this PHP course.'), "<br>";

    // How many words in a string
    echo "Words: ", str_word_count('I really like this PHP course.'), "<br>";

    // Reverse a string
    echo "Reversed: ", strrev('I really like this PHP course.'), "<br>";

    // Find the position of a substring
    echo "PHP starts in: ", strpos('I really like this PHP course.', 'PHP'), "<br>";

    echo "<hr>";

    // Repeat a string
    echo "Code 5 times: ", str_repeat('Code ', 5), "<br>";

    // Replace a substring
    $apple = "An apple a day will keep the doctor away.";
    echo $apple, "<br>";
    echo str_replace('n apple', ' beer', $apple);

    ?>
    </h3>
</body>
</html>
```

Debugging Code

Examine the following code. Find and fix the errors.

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Debugging Episode 4</title>
    <style> body{margin: 30px;} </style>
</head>
<body>
    <?php
    echo "Episode 4 was "All about Strings" and I liked it";

    $msg  'My best friend in the whole world is Hallie'.
    echo $msg;

    echo 'I need \$100 to buy groceries for the week.';

    ?>
</body>
</html>
```

Lab Exercises

The following exercises are meant to help you practice your coding skills. They need to be submitted for you to get a certification. As stated before coding is not a spectator sport, the more you practice (and make mistakes) the more you will learn.

1. Create 5 string variables and initialize each with one of the following strings. Use the appropriate type of quotation marks, and if necessary, escape sequences.

- Robert said, “I studied hard for my physics test.”
- John asked, “When are we going to see that movie?”
- Matthew told his friend, ‘Your bike looks just like mine.’
- Brooke told her cousins, “You’re invited to my party.”
- Eric told his brother, “If you give me $5 I won’t tell mom.”

2. Given the following three variables:
   $h = “Hydrogen”;
   $o = “Oxygen”;
   $w = “Water”;
   Use them in an echo statement to obtain the following output:
   Two molecules of Hydrogen and one molecule of Oxygen make one molecule of Water.

3. Use the strlen( ) function to find the length of “Well done is better than well said.”

4. Use the str_word_count( ) function to find how many words there are in “Tell me and I forget. Teach

me and I remember. Involve me and I learn."

5. Use the strrev( ) function to reverse the following string "An investment in knowledge pays the best interest."

6. Use the strpos( ) function to find in which position the substring "test" begins in the string "I didn't fail the test, I just found 100 ways to do it wrong."