Built-in Functions

This lesson discusses the in-built functions used for performing various string operations.

We'll cover the following

Extracting or Replacing Substrings

Single characters can be extracted using *array* (square brace) *syntax* as well as *curly brace* syntax. These two syntaxes will only return a single character from the string. If we need more than one character, we have to use the in-built `substr` function.

> **Note:** Strings, like everything in PHP, are 0-indexed, i.e., their first character is at index 0.

```
<?php
$foo = 'Hello world';
echo $foo[6]; // returns 'w'
echo "\n";
echo $foo{6}; // also returns 'w'
echo "\n";
echo substr($foo, 6, 1); // also returns 'w'
echo "\n";
echo substr($foo, 6, 2); // returns 'wo'
?>
```

Finding Position of a Substring

Finding Position of a Substring

In PHP you can use the `strpos` method to get the position/occurrence of a substring in another string. If the substring does not exist, the `strpos` returns `false`.

```
<?php
echo "The occurence of hay is at position: ".strpos("haystack", "hay")."\n"; // int(0)
echo "The occurence of stack is at position: ".strpos("haystack", "stack")."\n"; // int(3)
?>
```

Replacing Characters in a String

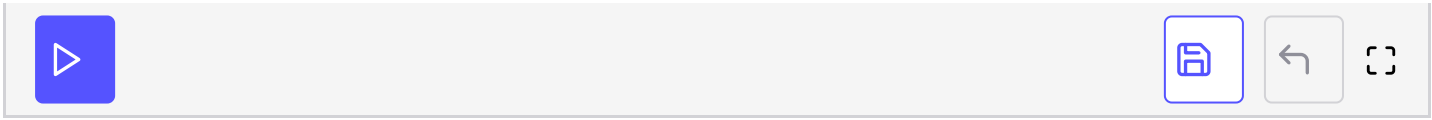
Strings can also be changed one character at a time using the same square brace and curly brace syntax. Replacing more than one character requires a function, `substr_replace`.

```
<?php
$foo = 'hello world';

$foo[6] = 'W'; // capitalizes the 'w' in 'hello world'
echo $foo;
echo "\n";

$foo{0} = 'H'; // capitalizes the 'h' in 'hello world'
echo $foo;
echo "\n";

$bar = substr_replace($foo, '!', 11, 1); // results in $bar = 'Hello World!'
echo $bar;
echo "\n";

$bar = substr_replace($foo, 'Whi', 6, 2); // results in 'Hello Whirld'
// Note that the replacement string need not be the same length as the substring replaced
echo $bar;
echo "\n";
?>
```

> **Note:** The substr_replace function does not change the actual string. It just returns the new string that would've been made after doing the replacement under discussion.

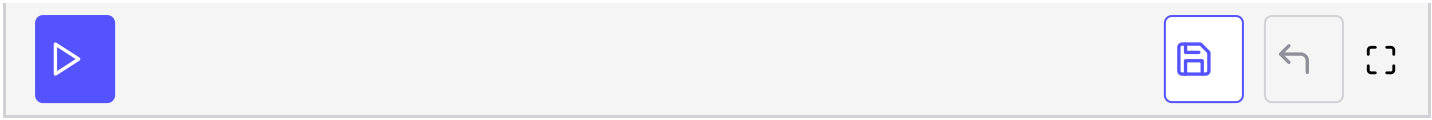
You can do this using the `str_replace()` method which essentially replaces all occurrences of the search text within the target string.

```
<?php
$my_str = 'If the facts do not fit the theory, change the facts.';

// replaces "facts" with "truth" and displays new string
echo str_replace("facts", "truth", $my_str);
?>
```



You can optionally pass the fourth argument to the `str_replace()` function to know how many times the string replacements were performed, like so:

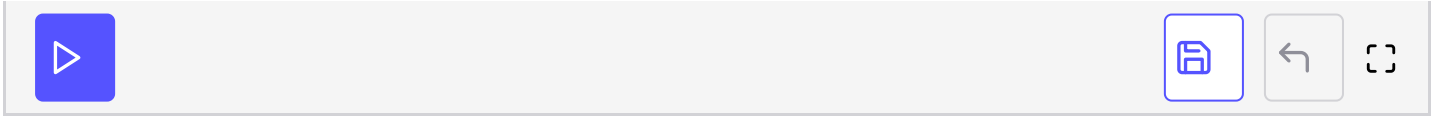
```
<?php
$my_str = 'If the facts do not fit the theory, change the facts.';

// Perform string replacement
str_replace("facts", "truth", $my_str, $count);

// Display number of replacements performed
echo "The text was replaced $count times.";
?>
```



Calculating the Length of a String

The `strlen()` function is used to calculate the number of characters inside a string. It also includes the blank spaces inside the string.

```
<?php
$my_str = 'Welcome to Educative!';
echo strlen($my_str);
?>
```



Counting the Number of Words in a String

The `str_word_count()` function counts the number of words in a string.

```php
<?php
$my_str = 'The quick brown fox jumps over the lazy dog.';

echo str_word_count($my_str);
?>
```

Reversing a String

The `strrev()` function in PHP can be used to reverse a string.

```php
<?php
$my_str = 'You can do anything, but not everything.';

// Display reversed string
echo strrev($my_str);
?>
```

In the next lesson try attempting the challenge in order to test your skills.