

PHP String

Last Updated Thursday, 27 March 2008

A string variable is used to store and manipulate a piece of text. Strings in PHPString variables are used for values that contains character strings. In this tutorial we are going to look at some of the most common functions and operators used to manipulate strings in PHP. String Creation Before you can use a string you have to create it! A string can be used directly in a function or it can be stored in a variable. Below we create the exact same string twice: first storing it into a variable and in the second case we place the string directly into a function. After we create a string we can manipulate it. A string can be used directly in a function or it can be stored in a variable. Below, the PHP script assigns the string "Hello World" to a string variable called \$txt:

```
<?php
$t="Hello World";
echo $t;
```

?>The output of the code above will be: Hello World String Creation Single Quotes Thus far we have created strings using double-quotes, but it is just as correct to create a string using single-quotes, otherwise known as apostrophes. PHP Code:

```
$string = 'Unlock your potential!';
echo 'Unlock your potential!';
```

If you want to use a single-quote within the string you have to escape the single-quote with a backslash \ . Like this: \! PHP Code: echo ' It\'s Neat!'; String Creation Double-Quotes We have used double-quotes and will continue to use them as the primary method for forming strings. Double-quotes allow for many special escaped characters to be used that you cannot do with a single-quote string. Once again, a backslash is used to escape a character. PHP Code:

```
$newline = "A newline is \n";
$return = "A carriage return is \r";
$tab = "A tab is \t";
$dollar = "A dollar sign is \$";
```

\$doublequote = "A double-quote is \\""; Note: If you try to escape a character that doesn't need to be, such as an apostrophe, then the backslash will show up when you output the string. These escaped characters are not very useful for outputting to a web page because HTML ignore extra white space. A tab, newline, and carriage return are all examples of extra (ignorable) white space. However, when writing to a file that may be read by human eyes these escaped characters are a valuable tool! PHP - String Creation Heredoc The two methods above are the conventional way to create strings in most programming languages. PHP introduces a more robust string creation tool called heredoc that lets the programmer create multi-line strings without using quotations. However, creating a string using heredoc is more difficult and can lead to problems if you do not correctly code your string! Here's how to do it: PHP Code:

```
$string =
<<<TEST
Unlock your potential!
TEST;
```

There are a few very important things to remember when using heredoc. Use <<< and some identifier that you choose to begin the heredoc. In this example we chose TEST as our identifier. Repeat the identifier followed by a semicolon to end the heredoc string creation. In this example that was TEST; The closing sequence TEST; must occur on a line by itself and cannot be indented! Another thing to note is that when you output this multi-line string to a web page, it will not span multiple lines because we did not have any
 tags contained inside our string! Here is the output made from the code above.

Display: Unlock your potential! Once again, take great care in following the heredoc creation guidelines to avoid any headaches. The Concatenation Operator There is only one string operator in PHP. The concatenation operator (.) is used to put two string values together. To concatenate two variables together, use the dot (.) operator: <?php \$t1="Hello World";

```
$t2="1234";
```

echo \$t1 . " " . \$t2; ?> The output of the code above will be: Hello World 1234 If we look at the code above you see that we used the concatenation operator two times. This is because we had to insert a third string. Between the two string variables we added a string with a single character, an empty space, to separate the two variables. Using the strlen() function

The strlen() function is used to find the length of a string. Let's find the length of our string "Hello world!": <?php echo strlen("Hello world!"); ?> The output of the code above will be: 12

The length of a string is often used in loops or other functions, when it is important to know when the string ends. (i.e. in a loop, we would want to stop the loop after the last character in the string) Using the strpos() function The strpos() function is used to search for a string or character within a string. If a match is found in the string, this function will return the position of the first match. If no match is found, it will return FALSE. Let's see if we can find the string "world" in our string: <?php echo strpos("Hello world!","world"); ?> The output of the code above will be: 6 As you see the position of the string "world" in our string is position 6. The reason that it is 6, and not 7, is that the first position in the string is 0, and not 1.