PHP addslashes() Function

Definition and Usage

The addslashes() function returns a string with backslashes in front of predefined characters.

The predefined characters are:

- single quote (')
- double quote (")
- backslash (\)
- NULL

Syntax

```
addslashes(string)
```

Parameter	Description
string	Required. Specifies the string to check

Example

In this example we will add backslashes to the predefined characters in a string:

```
<?php
$str = "Who's Kai Jim?";
echo $str . " This is not safe in a database query.<br />";
echo addslashes($str) . " This is safe in a database query.";
?>
```

The output of the code above will be:

```
Who's Kai Jim? This is not safe in a database query.
Who\'s Kai Jim? This is safe in a database query.
```

PHP stripslashes() Function

Definition and Usage

The stripslashes() function removes backslashes added by the addslashes() function.

Syntax

```
stripslashes(string)
```

Parameter	Description
string	Required. Specifies the string to check

Tips and Notes

Tip: This function can be used to clean up data retrieved from a database or from an HTML form.

Example

```
<?php
echo stripslashes("Who\'s Kai Jim?");
?>
```

The output of the code above will be:

```
Who's Kai Jim?
```

PHP chunk_split() Function

Definition and Usage

The chunk_split() function splits a string into a series of smaller parts.

Syntax

```
chunk_split(string,length,end)
```

Parameter	Description
string	Required. Specifies the string to split
length	Optional. A number that defines the length of the chunks. Default is 76
end	Optional. A string that defines what to place at the end of each chunk. Default is \r\n

Example 1

In this example we will split the string after each character and add a "." after each split:

```
<?php
$str = "Hello world!";
echo chunk_split($str,1,".");
?>
```

The output of the code above will be:

```
H.e.l.l.o. .w.o.r.l.d.!.
```

Example 2

In this example we will split the string after the sixth character and add "..." after each split:

```
<?php
$str = "Hello world!";
echo chunk_split($str,6,"...");
?>
```

The output of the code above will be:

```
Hello ...world!...
```

PHP explode() Function

Definition and Usage

The explode() function breaks a string into an array.

Syntax

```
explode(separator,string,limit)
```

Parameter	Description
separator	Required. Specifies where to break the string
string	Required. The string to split
limit	Optional. Specifies the maximum number of array elements to return

Tips and Notes

Note: Separator cannot be an empty string.

Example

In this example we will break a string to an array:

```
<?php
$str = "Hello world. It's a beautiful day.";
print_r (explode(" ",$str));
?>
```

The output of the code above will be:

```
Array
(
[0] => Hello
[1] => world.
[2] => It's
[3] => a
[4] => beautiful
[5] => day.
)
```

PHP sprintf() Function

Definition and Usage

The sprintf() function writes a formatted string to a variable.

The arg1, arg2, ++ parameters will be inserted at percent (%) signs in the main string. This function works "step-by-step". At the first % sign, arg1 is inserted, at the second % sign, arg2 is inserted, etc.

Syntax

```
sprintf(format,arg1,arg2,arg++)
```

Parameter	Description
format	Required. Specifies the string and how to format the variables in it. Possible format values: • %% - Returns a percent sign • %b - Binary number • %c - The character according to the ASCII value • %d - Signed decimal number • %e - Scientific notation (e.g. 1.2e+2) • %u - Unsigned decimal number • %f - Floating-point number (local settings aware) • %F - Floating-point number (not local settings aware) • %o - Octal number • %s - String • %x - Hexadecimal number (lowercase letters) • %X - Hexadecimal number (uppercase letters)

Example 1

```
<?php
$str = "Hello";
$number = 123;
$txt = sprintf("%s world. Day number %u",$str,$number);
echo $txt;
?>
```

The output of the code above will be:

```
Hello world. Day number 123
```

Example 2

```
<?php
$number = 123;
$txt = sprintf("%f",$number);
echo $txt;
?>
```

The output of the code above will be:

```
123.000000
```

Example 3

Use of placeholders:

```
<?php
$number = 123;
$txt = sprintf("With 2 decimals: %1\$.2f
<br />With no decimals: %1\$u",$number);
echo $txt;
?>
```

The output of the code above will be:

```
With 2 decimals: 123.00
With no decimals: 123
```

PHP str_pad() Function

Definition and Usage

The str_pad() function pads a string to a new length.

Syntax

```
str_pad(string,length,pad_string,pad_type)
```

Parameter	Description
string	Required. Specifies the string to pad
length	Required. Specifies the new string length. If this value is less than the original length of the string, nothing will be done
pad_string	Optional. Specifies the string to use for padding. Default is whitespace
pad_type	Optional. Specifies what side to pad the string. Possible values: • STR_PAD_BOTH - Pad to both sides of the string. If not an even number, the right side gets the extra padding • STR_PAD_LEFT - Pad to the left side of the string • STR_PAD_RIGHT - Pad to the right side of the string. This is default

Example 1

```
<?php
$str = "Hello World";
echo str_pad($str,20,".",STR_PAD_LEFT);
?>
```

The output of the code above will be:

```
.........Hello World
```

Example 2

```
<?php
$str = "Hello World";
echo str_pad($str,20,".:",STR_PAD_BOTH);
?>
```

The output of the code above will be:

```
.:.:Hello World.:.:.
```

PHP str_repeat() Function

Definition and Usage

The str_repeat() function repeats a string a specified number of times.

Syntax

```
str_repeat(string,repeat)
```

Parameter	Description
string	Required. Specifies the string to repeat
repeat	Required. Specifies the number of times the string will be repeated. Must be greater or equal to 0

Example

```
<?php
echo str_repeat(".",13);
?>
```

The output of the code above will be:

```
.............
```

PHP str_replace() Function

Definition and Usage

The str_replace() function replaces some characters with some other characters in a string. This function is case-sensitive. Use str_ireplace() to perform a case-insensitive search.

Syntax

```
str_replace(find,replace,string,count)
```

Parameter	Description
find	Required. Specifies the value to find
replace	Required. Specifies the value to replace the value in *find*
string	Required. Specifies the string to be searched

Example 1

```
<?php
echo str_replace("world","Peter","Hello world!");
?>
```

The output of the code above will be:

```
Hello Peter!
```

Example 2

In this example we will demonstrate str_replace() with an array and a count variable:

```
<?php
$arr = array("blue","red","green","yellow");
print_r(str_replace("red","pink",$arr,$i));
echo "Replacements: $i";
?>
```

The output of the code above will be:

```
Array
(
[0] => blue
[1] => pink
[2] => green
[3] => yellow
)
Replacements: 1
```

PHP str_shuffle() Function

Definition and Usage

The str_shuffle() function randomly shuffles all the characters of a string.

Syntax

```
str_shuffle(string)
```

Parameter	Description
string	Required. Specifies the string to shuffle

Example

```
<?php
echo str_shuffle("Hello World");
?>
```

The output of the code above could be:

```
H leooWlrld
```

PHP str_split() Function

Definition and Usage

The str_split() function splits a string into an array.

Syntax

```
str_split(string,length)
```

Parameter	Description
string	Required. Specifies the string to split
length	Optional. Specifies the length of each array element. Default is 1

Example 1

```
<?php
print_r(str_split("Hello"));
?>
```

The output of the code above will be:

```
Array
(
[0] => H
[1] => e
[2] => l
[3] => l
[4] => o
)
```

Example 2

```
<?php
print_r(str_split("Hello",3));
?>
```

The output of the code above will be:

```
Array
(
[0] => Hel
[1] => lo
)
```

PHP str_word_count() Function

Definition and Usage

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

Syntax

```
str_word_count(string,return,char)
```

Parameter	Description
string	Required. Specifies the string to check

Example 1

```
<?php
echo str_word_count("Hello world!");
?>
```

The output of the code above will be:

```
2
```

PHP strcmp() Function

Definition and Usage

The strcmp() function compares two strings.

This function returns:

- 0 - if the two strings are equal
- <0 - if string1 is less than string2
- >0 - if string1 is greater than string2

Syntax

```
strcmp(string1,string2)
```

Parameter	Description
string1	Required. Specifies the first string to compare
string2	Required. Specifies the second string to compare

Tips and Notes

Note: The strcmp() function is binary safe and case-sensitive.

Example

```
<?php
echo strcmp("Hello world!","Hello world!");
?>
```

The output of the code above will be:

```
0
```

PHP strip_tags() Function

Definition and Usage

The strip_tags() function strips a string from HTML, XML, and PHP tags. HTML comments are always stripped. This cannot be changed with the allow parameter.

Syntax

```
strip_tags(string,allow)
```

Parameter	Description
string	Required. Specifies the string to check
allow	Optional. Specifies allowable tags. These tags will not be removed

Example 1

```
<?php
echo strip_tags("Hello <b>world!</b>");
?>
```

The output of the code above will be:

```
Hello world!
```

Example 2

```
<?php
echo strip_tags("Hello <b><i>world!</i></b>","<b>");
?>
```

The output of the code above will be:

```
Hello world!
```

PHP strpos() Function

Definition and Usage

The strpos() function returns the position of the first occurrence of a string inside another string.

If the string is not found, this function returns FALSE.

Syntax

```
strpos(string,find,start)
```

Parameter	Description
string	Required. Specifies the string to search
find	Required. Specifies the string to find
start	Optional. Specifies where to begin the search

Tips and Notes

Note: The strpos() function is case-sensitive.

Example

```
<?php
echo strpos("Hello world!","wo");
?>
```

The output of the code above will be:

```
6
```

PHP stripos() Function

Definition and Usage

The stripos() function returns the position of the first occurrence of a string inside another string.

If the string is not found, this function returns FALSE.

Syntax

```
stripos(string,find,start)
```

Parameter	Description
string	Required. Specifies the string to search
find	Required. Specifies the string to find
start	Optional. Specifies where to begin the search

Tips and Notes

Note: The stripos() function is case-insensitive.

Example

```
<?php
echo stripos("Hello world!","WO");
?>
```

The output of the code above will be:

```
6
```

PHP substr() Function

Definition and Usage

The substr() function returns a part of a string.

Syntax

```
substr(string,start,length)
```

Parameter	Description
string	Required. Specifies the string to return a part of
start	Required. Specifies where to start in the string • A positive number - Start at a specified position in the string • A negative number - Start at a specified position from the end of the string • 0 - Start at the first character in string
length	Optional. Specifies the length of the returned string. Default is to the end of the string. • A positive number - The length to be returned from the start parameter • Negative number - The length to be returned from the end of the string

Example 1

```
<?php
echo substr("Hello world!",6);
?>
```

The output of the code above will be:

```
world!
```

Example 2

```
<?php
echo substr("Hello world!",6,5);
?>
```

The output of the code above will be:

```
world
```

PHP strtolower() Function

Definition and Usage

The strtolower() function converts a string to lowercase.

Syntax

```
strtolower(string)
```

Parameter	Description
string	Required. Specifies the string to convert

Example

```
<?php
echo strtolower("Hello WORLD.");
?>
```

The output of the code above will be:

```
hello world.
```

PHP strtoupper() Function

Definition and Usage

The strtoupper() function converts a string to uppercase.

Syntax

```
strtoupper(string)
```

Parameter	Description
string	Required. Specifies the string to convert

Example

```
<?php
echo strtoupper("Hello WORLD!");
?>
```

The output of the code above will be:

```
HELLO WORLD!
```

PHP trim() Function

Definition and Usage

The trim() function removes whitespaces and other predefined characters from both sides of a string.

Syntax

```
trim(string,charlist)
```

Parameter	Description
string	Required. Specifies the string to check
charlist	Optional. Specifies which characters to remove from the string. If omitted, all of the following characters are removed: • "\0" - NULL • "\t" - tab • "\n" - new line • "\x0B" - vertical tab • "\r" - carriage return • " " - ordinary white space

Example 1

```
<html>
<body>
<?php
$str = "     Hello World!     ";
echo "Without trim: " . $str;
echo "<br />";
echo "With trim: " . trim($str);
?>
<body>
<html>
```

The browser output of the code above will be:

```
Without trim: Hello World!
With trim: Hello World!
```

PHP substr_replace() Function

Definition and Usage

The substr_replace() function replaces a part of a string with another string.

Syntax

```
substr_replace(string,replacement,start,length)
```

Parameter	Description
string	Required. Specifies the string to check
replacement	Required. Specifies the string to insert
start	Required. Specifies where to start replacing in the string • A positive number - Start replacing at the specified position in the string • Negative number - Start replacing at the specified position from the end of the string • 0 - Start replacing at the first character in the string
length	Optional. Specifies how many characters should be replaced. Default is the same length as the string. • A positive number - The length of string to be replaced • A negative number - How many characters should be left at end of string after replacing • 0 - Insert instead of replace

Tips and Notes

Note: If start is a negative number and length is less than or equal to start, length becomes 0.

Example

```
<?php
echo substr_replace("Hello world","earth",6);
?>
```

The output of the code above will be:

```
Hello earth
```

PHP substr_count() Function

Definition and Usage

The substr_count() function counts the number of times a substring occurs in a string.

Syntax

```
substr_count(string,substring,start,length)
```

Parameter	Description
string	Required. Specifies the string to check
substring	Required. Specifies the string to search for
start	Optional. Specifies where in string to start searching
length	Optional. Specifies the length of the search

Example

```
<?php
echo substr_count("Hello world. The world is nice","world");
?>
```

The output of the code above will be:

```
2
```

PHP substr_compare() Function

Definition and Usage

The substr_compare() function compares two strings from a specified start position.

This function returns:

- 0 - if the two strings are equal
- <0 - if string1 (from startpos) is less than string2
- >0 - if string1 (from startpos) is greater than string2

If length is equal or greater than length of string1, this function returns FALSE.

Syntax

```
substr_compare(string1,string2,startpos,length,case)
```

Parameter	Description
string1	Required. Specifies the first string to compare
string2	Required. Specifies the second string to compare
startpos	Required. Specifies where to start comparing in string1
length	Optional. Specifies how much of string1 to compare
case	Optional. Specifies whether or not to perform a case-sensitive compare. Default is FALSE (case-sensitive)

Example 1

```
<?php
echo substr_compare("Hello world","Hello world",0);
?>
```

The output of the code above will be:

```
0
```

Example 2

```
<?php
echo substr_compare("Hello world","WORLD",6,TRUE);
?>
```

The output of the code above will be:

```
0
```