Chapter 3: PHP String Functions

1. String Functions

echo()
The echo() function outputs one or more strings.

```
<?php
echo "Hello world!";
?>
```

print_r ()
print_r is used to return an array in a human readable form in PHP

```
<?php
$numbers = array( 1, 2, 3, 4, 5);
print_r($numbers);
?>
```

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

```
<html> <body>
<?php
$str = 'one,two,three,four';

// zero limit
print_r(explode(',',$str,0));
print "<br>";

// positive limit
print_r(explode(',',$str,2));
print "<br>";

// negative limit
print_r(explode(',',$str,-1));
?>

</body> </html>
```

implode()
Returns a string from the elements of an array. The implode() function returns a string from the elements of an array.

1.
```
<?php
$arr = array('Hello','World!','Beautiful','Day!');
$a=implode(" ",$arr);
echo $a."<br>";
echo $arr[0];
?>
```

2.
```
<?php
$arr = array('Hello','World!','Beautiful','Day!');
echo implode(" ",$arr)."<br>";
echo implode("+",$arr)."<br>";
```

```
echo implode("-",$arr)."<br>";
echo implode("X",$arr);
?>
```

join()
The join() function is an alias of the implode() function.
Join array elements with a string:

```
1.
<?php
$arr = array('Hello','World!','Beautiful','Day!');
echo join(" ",$arr);
?>

2.
<?php
$arr = array('Hello','World!','Beautiful','Day!');
echo join(" ",$arr)."<br>";
echo join("+",$arr)."<br>";
echo join("-",$arr)."<br>";
echo join("X",$arr);
?>
```

lcfirst()
The lcfirst() function converts the first character of a string to lowercase.

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

ucfirst()
The ucfirst() function converts the first character of a string to uppercase.

```
<?php
echo ucfirst("hello world!");
?>
```

ucwords()
The ucwords() function converts the first character of each word in a string to uppercase

```
<?php
echo ucwords("hello world");
?>
```

strtoupper()
The strtoupper() function converts a string to uppercase.

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

strtolower()
The strtolower() function converts a string to lowercase. Convert all characters to lowercase:

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

ltrim()
The ltrim() function removes whitespace or other predefined characters from the left side of a string.

```
<?php
$str = "Hello World!";
echo $str . "<br>";
echo ltrim($str,"Hello");
?>
```

rtrim()

The rtrim() function removes whitespace or other predefined characters from the right side of a string.

```
<?php
$str = "Hello World!";
echo $str . "<br>";
echo rtrim($str,"World!");
?>
```

trim()

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

```
<?php
$str = "Hello World!";
echo $str . "<br>";
echo trim($str,"Hed!");
?>
```

md5()

The md5() function calculates the MD5 hash of a string. The md5() function uses the RSA Data Security, Inc. MD5 Message-Digest Algorithm.

```
1.
<?php
$str = "Hello";
echo md5($str);
?>

2.
<?php
$filename = "test.txt";
$md5file = md5_file($filename);
echo $md5file;
?>
```

sha1()

The sha1() function calculates the SHA-1 hash of a string.

```
<?php
$str = "Hello";
echo sha1($str);
?>

<?php
$filename = "test.txt";
$sha1file = sha1_file($filename);
echo $sha1file;
echo
?>
```

money_format()

The money_format() function returns a string formatted as a currency string. This function inserts a formatted number where there is a percent (%) sign in the main string.

```
<?php
$number = 1234.56;
setlocale(LC_MONETARY,"en_US");
echo money_format("The price is %i", $number);
?>
```

print()

The print() function outputs one or more strings.

```
<?php
print "Hello world!";
?>
```

printf()

The printf() function outputs a formatted string.

```
<?php
$number = 9;
$str = "Beijing";
printf("There are %u million bicycles in %s.",$number,$str);
?>
```

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 (negative, zero or positive)
- %e - Scientific notation using a lowercase (e.g. 1.2e+2)
- %E - Scientific notation using a uppercase (e.g. 1.2E+2)
- %u - Unsigned decimal number (equal to or greather than zero)
- %f - Floating-point number (local settings aware)
- %F - Floating-point number (not local settings aware)
- %g - shorter of %e and %f
- %G - shorter of %E and %f
- %o - Octal number
- %s - String
- %x - Hexadecimal number (lowercase letters)
- %X - Hexadecimal number (uppercase letters)

sprintf()

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

```
<?php
$number = 9;
$str = "Beijing";
$txt = sprintf("There are %u million bicycles in %s.",$number,$str);
echo $txt;
?>
```

fprintf()

The fprintf() function writes a formatted string to a specified output stream (example: file or database).

```
<?php
$tm = 9;
$str= "Morning";
$file = fopen("test.txt","w");
fprintf($file,"Its $tm :00 AM So "." Good ".$str);
```

```
?>
```

str_replace()

The str_replace() function replaces some characters with some other characters in a string.
This function works by the following rules:

- If the string to be searched is an array, it returns an array
- If the string to be searched is an array, find and replace is performed with every array element
- If both find and replace are arrays, and replace has fewer elements than find, an empty string will be used as replace
- If find is an array and replace is a string, the replace string will be used for every find value

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

str_ireplace()

The str_ireplace() function replaces some characters with some other characters in a string.
This function works by the following rules:

- If the string to be searched is an array, it returns an array
- If the string to be searched is an array, find and replace is performed with every array element
- If both find and replace are arrays, and replace has fewer elements than find, an empty string will be used as replace
- If find is an array and replace is a string, the replace string will be used for every find value

Replace the characters "WORLD" (case-insensitive) in the string "Hello world!" with "Peter":

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

str_shuffle()

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

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

str_split()

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

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

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

str_word_count()

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

```
<?php
echo str_word_count("Hello world!");
?>
<?php
print_r(str_word_count("Hello world!",1));
?>
```

strcasecmp()

The strcasecmp() function compares two strings. If this function returns 0, the two strings are equal.

```
<?php
echo strcasecmp("Hello world!","HELLO WORLD!");
?>
```

strcmp()

The strcmp() function compares two strings.

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

strchr()

The strchr() function searches for the first occurrence of a string inside another string.

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

stripos() ot strpos()

The stripos() ot strpos() function finds the position of the first occurrence of a string inside another string.

```
<?php
echo stripos("I love php, I love php too!","PHP");
?>
```

The strrpos() function finds the position of the last occurrence of a string inside another string.

```
<?php
echo strrpos("I love php, I love php too! php","php");
?>

<?php
echo strrpos("I love php, I love php too! Php I like php","php",20);
?>
```

strlen()

The strlen() function returns the length of a string.

```
<?php
echo strlen("Hello");
?>
```

strpbrk()

The strpbrk() function searches a string for any of the specified characters.

```
<?php
echo strpbrk("Hello world!","W");
echo "<br>";
echo strpbrk("Hello world!","w");
?>
```

strrev()

The strrev() function reverses a string.

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

strtok()

The strtok() function splits a string into smaller strings (tokens).

```
<?php
$string = "Hello world. Beautiful day today.";
$token = strtok($string, " ");
while ($token !== false)
    {
    echo "$token<br>";
    $token = strtok(" ");
    }
?>
```

strtr()

The strtr() function translates certain characters in a string.

```
<?php
echo strtr("Hilla Warld","ia","eo");
?>
```

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

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

substr_count()

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

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

wordwrap()

The wordwrap() function wraps a string into new lines when it reaches a specific length.

```
<?php
$str = "An example of a long word is: Supercalifragulistic";
echo wordwrap($str,15,"<br>\n");
?>
```

2. PHP Functions

PHP functions are similar to other programming languages. A function is a piece of code which takes one more input in the form of parameter and does some processing and returns a value.
You already have seen many functions like **fopen()** and **fread()** etc. They are built-in functions but PHP gives you option to create your own functions as well.
There are two parts which should be clear to you:

- Creating a PHP Function
- Calling a PHP Function

In fact you hardly need to create your own PHP function because there are already more than 1000 of built-in library functions created for different area and you just need to call them according to your requirement.

Creating PHP Function:

Its very easy to create your own PHP function. Suppose you want to create a PHP function which will simply write a simple message on your browser when you will call it. Following example creates a function called writeMessage() and then calls it just after creating it.
Note that while creating a function its name should start with keyword **function** and all the PHP code should be put inside { and } braces as shown in the following example below:

```
<html>
<head>
<title>Writing PHP Function</title>
</head>
<body>

<?php
/* Defining a PHP Function */
function writeMessage()
{
  echo "You are really a nice person, Have a nice time!";
}
/* Calling a PHP Function */
writeMessage();
?>
</body>
</html>
```

This will display following result:

```
You are really a nice person, Have a nice time!
```

PHP Functions with Parameters:

PHP gives you option to pass your parameters inside a function. You can pass as many as parameters your like. These parameters work like variables inside your function. Following example takes two integer parameters and add them together and then print them.

```
<html>
<head>
<title>Writing PHP Function with Parameters</title>
</head>
<body>

<?php
function addFunction($num1, $num2)
{
  $sum = $num1 + $num2;
  echo "Sum of the two numbers is : $sum";
}
addFunction(10, 20);
?>
</body>
</html>
```

This will display following result:

```
Sum of the two numbers is : 30
```

Passing Arguments by Reference:

It is possible to pass arguments to functions by reference. This means that a reference to the variable is manipulated by the function rather than a copy of the variable's value.

Any changes made to an argument in these cases will change the value of the original variable. You can pass an argument by reference by adding an ampersand to the variable name in either the function call or the function definition.

Following example depicts both the cases.

```
<html>
<head>
<title>Passing Argument by Reference</title>
</head>
<body>
<?php
function addFive($num)
{
   $num += 5;
}

function addSix(&$num)
{
   $num += 6;
}
$orignum = 10;
addFive( &$orignum );
echo "Original Value is $orignum<br />";
addSix( $orignum );
echo "Original Value is $orignum<br />";
?>
</body>
</html>
```

This will display following result:

```
Original Value is 15
Original Value is 21
```

PHP Functions retruning value:

A function can return a value using the **return** statement in conjunction with a value or object. return stops the execution of the function and sends the value back to the calling code.

You can return more than one value from a function using **return array(1,2,3,4)**.

Following example takes two integer parameters and add them together and then returns their sum to the calling program. Note that **return** keyword is used to return a value from a function.

```
<html>
<head>
<title>Writing PHP Function which returns value</title>
</head>
<body>

<?php
function addFunction($num1, $num2)
{
  $sum = $num1 + $num2;
  return $sum;
}
$return_value = addFunction(10, 20);
echo "Returned value from the function : $return_value
?>
</body>
</html>
```

This will display following result:

```
Returned value from the function : 30
```

Setting Default Values for Function Parameters:

You can set a parameter to have a default value if the function's caller doesn't pass it.
Following function prints NULL in case use does not pass any value to this function.

```
<html>
<head>
<title>Writing PHP Function which returns value</title>
</head>
<body>

<?php
function printMe($param = NULL)
{
   print $param;
}
printMe("This is test");
printMe();
?>

</body>
</html>
```

This will produce following result:

```
This is test
```

Dynamic Function Calls:

It is possible to assign function names as strings to variables and then treat these variables exactly as you would the function name itself. Following example depicts this behaviour.

```
<html>
<head>
<title>Dynamic Function Calls</title>
</head>
<body>
<?php
function sayHello()
{
   echo "Hello<br />";
}
$function_holder = "sayHello";
$function_holder();
?>
</body>
</html>
```

This will display following result:

```
Hello
```

3. Mathematical Functions

PHP abs() Function

Definition and Usage

The abs() function returns the absolute value of a number.

Syntax

abs(x)

Parameter	Description
x	Required. A number. If the number is of type float, the return type is also float, otherwise it is integer

Example

```
<?php
echo(abs(6.7) . "<br />");
echo(abs(-3) . "<br />");
echo(abs(3));
?>
```

The output of the code above will be:

6.7
3
3

PHP ceil() Function

Definition and Usage

The ceil() function returns the value of a number rounded UPWARDS to the nearest integer.

Syntax

ceil(x)

Parameter	Description
x	Required. A number

Example

In the following example we will use the ceil() function on different numbers:

```
<?php
echo(ceil(0.60) . "<br />");
echo(ceil(0.40) . "<br />");
echo(ceil(5) . "<br />");
echo(ceil(5.1) . "<br />");
echo(ceil(-5.1) . "<br />");
echo(ceil(-5.9))
?>
```

The output of the code above will be:

1
1
5
6
-5
-5

PHP cos() Function

Definition and Usage

The cos() function returns the cosine of a number.

Syntax

cos(x)

Parameter	Description
x	Required. A number

Tips and Notes

Note: The cos() function returns a numeric value between -1 and 1, which represents the cosine of the angle.

Example

In the following example we will return the cosine of different numbers:

```
<?php
echo(cos(3) . "<br />");
echo(cos(-3) . "<br />");
echo(cos(0) . "<br />");
echo(cos(M_PI) . "<br />");
echo(cos(2*M_PI))
?>
```

The output of the code above will be:

-0.9899924966004454
-0.9899924966004454
1
-1
1

PHP exp() Function

Definition and Usage

The exp() function returns the value of E^x, where E is Euler's constant (approximately 2.7183) and x is the number passed to it.

Syntax

exp(x)

Parameter	Description
x	Required. A number

Tips and Notes

Tip: E is the Euler's constant, which is the base of natural logarithms (approximately 2.7183).

Example

In the following example we will use the exp() function on different numbers:

```
<?php
echo(exp(1) . "<br />");
echo(exp(-1) . "<br />");
?>
```

The output of the code above will be:

```
2.718281828459045
0.36787944117144233
```

PHP floor() Function

Definition and Usage

The floor() function returns the value of a number rounded DOWNWARDS to the nearest integer.

Syntax

```
floor(x)
```

Parameter	Description
x	Required. A number

Example

In this example we will use the floor() function on different numbers:

```
<?php
echo(floor(0.60) . "<br />");
echo(floor(0.40) . "<br />");
echo(floor(5) . "<br />");
echo(floor(5.1) . "<br />");
echo(floor(-5.1) . "<br />");
echo(floor(-5.9))
?>
```

The output of the code above will be:

```
0
0
5
5
-6
-6
```

PHP max() Function

Definition and Usage

The max() function returns the number with the highest value of two specified numbers.

Syntax

max(x,y)

Parameter	Description
x	Required. A number
y	Required. A number

Example

In this example we will show how to use max() to return the number with the highest value of two specified numbers:

```
<?php
echo(max(5,7) . "<br />");
echo(max(-3,5) . "<br />");
echo(max(-3,-5) . "<br />");
echo(max(7.25,7.30))
?>
```

The output of the code above will be:

```
7
5
-3
7.3
```

PHP min() Function

Definition and Usage

The min() function returns the number with the lowest value of two specified numbers.

Syntax

min(x,y)

Parameter	Description
x	Required. A number
y	Required. A number

Example

In this example we will show how to use min() to return the number with the lowest value of two specified numbers:

```
<?php
echo(min(5,7) . "<br />");
echo(min(-3,5) . "<br />");
echo(min(-3,-5) . "<br />");
echo(min(7.25,7.30))
?>
```

The output of the code above will be:

5
-3
-5
7.25

PHP pow() Function

Definition and Usage

The pow() function raises the first argument to the power of the second argument, and returns the result.

Syntax

pow(x,y)

Parameter	Description
x	Required. Specifies the number to be raised
y	Required. The power to which to raise the number

Example

```
<?php
echo pow(4,2) . "<br />";
echo pow(6,2) . "<br />";
echo pow(-6,2) . "<br />";
echo pow(-6,-2) . "<br />";
?>
```

The output of the code above will be:

16
36
36
0.0277777777778

PHP round() Function

Definition and Usage

The round() function rounds a number to the nearest integer.

Syntax

round(x,prec)

Parameter	Description
x	Required. The number to be round
prec	Optional. The number of digits after the decimal point

Example

In this example we will round different numbers with the round() function:

```
<?php
echo(round(0.60) . "<br />");
```

```
echo(round(0.50) . "<br />");
echo(round(0.49) . "<br />");
echo(round(-4.40) . "<br />");
echo(round(-4.60))
?>
```

The output of the code above will be:

```
1
1
0
-4
-5
```

PHP sqrt() Function

Definition and Usage

The sqrt() function returns the square root of a number.

Syntax

sqrt(x)

Parameter	Description
x	Required. A number

Tips and Notes

Note: The sqrt() function will return -1.#IND if the parameter x is a negative number.

Example

In this example we will get the square root of different numbers:

```
<?php
echo(sqrt(0) . "<br />");
echo(sqrt(1) . "<br />");
echo(sqrt(9) . "<br />");
echo(sqrt(0.64) . "<br />");
?>
```

The output of the code above will be:

```
0
1
3
0.8
```