Chapter 9

How to work with strings and numbers

Objectives

Applied

1. Use any of the functions and techniques that are presented in this chapter to work with strings.
2. Use any of the functions and techniques that are presented in this chapter to work with numbers.

Objectives (continued)

Knowledge

1. Describe the way variable substitution is used to assign a string to a variable.
2. Describe the way PHP escape sequences can be used to insert special characters into strings and how the htmlentities function can be used to display special characters correctly in a browser.
3. Describe these terms as they apply to a PHP string: length, substring, and position.
4. Describe the use of the PHP string functions that return string lengths or substrings, search for or replace characters in a string, modify a string, convert between strings and arrays, and compare two strings.

Objectives (continued)

Knowledge (continued)

5. Describe the PHP is_infinite and is_finite functions, and describe these PHP constants: PHP_INT_MAX, INF, and -INF.
6. Describe these PHP functions for working with numbers: max, min, pow, round, sqrt, and mt_rand.
7. Describe the use of the sprintf function for formatting strings and numbers.
8. Describe the use of type casting and the use of the intval and floatval functions.

Assign strings with single quotes

```
$language = 'PHP';
$message = 'Welcome to ' . $language;

$query = 'SELECT firstName, lastName
          FROM Users';
```

Assign strings with double quotes

Using variable substitution

```
$language = "PHP";
$message = "Welcome to $language";
```

Using braces with variable substitution

```
$count = 12;
$item = "flower";
$message1 = "You bought $count $items.";
$message2 = "You bought $count ${item}s.";
```

Within a heredoc, all new line characters, tabs, and spaces are included as part of the string except for the last new line character which precedes the closing heredoc.

Assigning a string with a heredoc

```
$language = 'PHP';
$message = <<<MESSAGE
The heredoc syntax allows you to build multi-line
strings in $language. Inside, it acts like a
double-quoted string and performs variable
substitution.
MESSAGE;
```

Assign a string with a nowdoc

```
$message = <<<'MESSAGE'
The nowdoc syntax also allows you to build multi-
line strings in PHP. However, no variable
substitution takes place inside the nowdoc string.
This is similar to single-quoted strings.
MESSAGE;
```

Escape sequences only used in some strings

```
\\      Use in all strings except nowdocs
\'      Use in single-quoted strings
\"      Use in double-quoted strings
```

Escape sequences used in double-quoted strings and heredocs. Not used in single-quoted strings or nowdocs.

```
\$      dollar sign
\n      new line
\t      tab
\r      carriage return
\f      form feed
\v      vertical tab
\ooo    octal character
\xhh    hexadecimal character
```

Escape sequences with single quotes

```
$dir = 'C:\\xampp\\php';
$name = 'Mike\'s Music Store';
$quote = "He said, \"It costs \$12.\"";
$comment1 = "This is a\nmulti-line string.";
$comment2 = 'Not a\nmulti-line string.';
```

The escape sequences for octal and hexadecimal values let you use any ASCII character in a string. But, browsers don't always display some characters correctly.

The htmlentities function corrects that:

```
htmlentities($str [, $quotes])
```

Examples of the htmlentities function

An example that doesn't use the htmlentities function

```
$copyright1 = "\xa9 2010";
echo $copyright1;                    //Displays  2010
```

An example that uses the htmlentities function

```
$copyright2 = htmlentities("\xa9 2010");
echo $copyright2;                    //Displays © 2010
```

A URL for a list of all PHP string functions

```
http://www.php.net/manual/en/ref.strings.php
```

Functions for working with string length and substrings

```
empty($str)
strlen($str)
substr($str, $i[, $len])
```

Code that determines if a string is empty

```
if (empty($first_name)) {
    $message = 'You must enter the first name.';
}
```

Code that gets the length of a string and two substrings

```
$name = 'Ray Harris';
$length = strlen($name);
$first_name = substr($name, 0, 3);
$last_name = substr($name, 4);
$last_name = substr($name, -6);
```

Code that formats a phone number in two ways

```
$phone = '5545556624';
$part1 = substr($phone, 0, 3);
$part2 = substr($phone, 3, 3);
$part3 = substr($phone, 6);
$format_1 = $part1 . '-' . $part2 . '-' . $part3;
$format_2 = '(' . $part1 . ') ' . $part2 . '-' . $part3;
```

Code that displays each letter in a string on a separate line

```
$input = 'JAN';
for ($i = 0; $i < strlen($input); $i++) {
    $vert_str .= substr($input, $i, 1);
    $vert_str .= '<br />';
}
```

Functions that search a string

```
strpos($str1, $str2[, $offset])
stripos($str1, $str2[, $offset])
strrpos($str1, $str2[, $offset])
strripos($str1, $str2[, $offset])
```

Code that searches a string for spaces

```
$name = 'Martin Van Buren';
$i = strpos($name, ' ');
$i = strpos($name, ' ', 7);
$i = strrpos($name, ' ');
```

Code that searches a string for a substring

```
$name = 'Martin Van Buren';
$i = strpos($name, 'Van');
$i = strpos($name, 'van');
$i = stripos($name, 'van');
$i = strripos($name, 'A');
```

Code that splits a string into two substrings

```
$name = 'Ray Harris';
$i = strpos($name, ' ');
if ($i === false) {
    $message = 'No spaces were found in the name.';
} else {
    $first_name = substr($name, 0, $i);
    $last_name = substr($name, $i+1);
}
```

Functions that replace part of a string

```
str_replace($str1, $new, $str2)
str_ireplace($str1, $new, $str2)
```

Code that replaces periods with dashes

```
$phone = '554.555.6624';
$phone = str_replace('.', '-', $phone);
```

Code that replaces one string with another string

```
$message = 'Hello Ray';
$message = str_ireplace('hello', 'Hi', $message);
```

Functions that modify strings

```
ltrim($str)
rtrim($str)
trim($str)
str_pad($str, $len [, $pad[, $type]])
lcfirst($str)
ucfirst($str)
ucwords($str)
strtolower($str)
strtoupper($str)
strrev($str)
str_shuffle($str)
str_repeat($str, $i)
```

Code that trims and pads a string

```
$name = '   ray harris   ';
$name = ltrim($name);
$name = rtrim($name);

$name = str_pad($name, 13);
$name = str_pad($name, 16,
        ' ', STR_PAD_LEFT);

$name = trim($name);
```

Code that works with capitalization

```
$name = ucfirst($name);
$name = lcfirst($name);
$name = ucwords($name);
$name = strtolower($name);
$name = strtoupper($name);
```

Code that changes the sequence of the characters

```
$name = strrev($name);
$name = str_shuffle($name);
```

Code that repeats a string

```
$sep = str_repeat('*', 10);
```

Functions that convert strings and arrays

```
explode($sep, $str)
implode($sep, $sa)
```

How to convert a string to an array

```
$names = 'Mike|Anne|Joel|Ray';
$names = explode('|', $names);
$name1 = $names[0];
$name2 = $names[1];
```

How to convert an array to a string

```
$names = implode('|', $names);
```

How to convert an array to a tab-delimited string

```
$names = implode('\t', $names);
```

Functions that convert between strings and ASCII integer values

```
chr($value)
ord($string)
```

How to convert an integer value to a character

```
$char = chr(65);      // $char is 'A'
$char = chr(66);      // $char is 'B'
```

How to convert a character to an integer value

```
$val = ord('A');      // $val is 65
$val = ord('B');      // $val is 66
$val = ord('Bike');   // $val is 66
```

Functions that compare two strings

```
strcmp($str1, $str2)
strcasecmp($str1, $str2)
strnatcmp($str1, $str2)
strnatcasecmp($str1, $str2)
```

How a case-sensitive comparison works

```
$result = strcmp('Anders', 'Zylka');      // $result = -1
$result = strcmp('Anders', 'zylka');      // $result = 1
$result = strcasecmp('Anders', 'zylka'); // $result = -25
```

How a “natural” number comparison works

```
$result = strcmp('img06', 'img10');       // $result = -1
$result = strcmp('img6', 'img10');        // $result = 1
$result = strnatcmp('img6', 'img10');     // $result = -1
```

How to compare two strings

```
$result = strnatcasecmp($name_1, $name_2);

if ($result < 0) {
    echo $name_1 . ' before ' . $name_2;
} else if ($result == 0) {
    echo $name_1 . ' matches ' . $name_2;
} else {
    echo $name_1 . ' after ' . $name_2;
}
```

How to assign a decimal value (base 10)

```
$number_1 = 42;
$number_2 = +72;
$number_3 = -13;
$number_4 = -(-39);
$number_5 = --39;    // Error
```

How to find the maximum and minimum integer values (base 10)

```
$max_int = PHP_INT_MAX;
$min_int = -1 * (PHP_INT_MAX + 1);
```

How to assign an octal value (base 8)

```
$octal_1 = 0251;      // Must begin with 0
$octal_2 = -0262;
```

How to assign a hexadecimal value (base 16)

```
$hex_1 = 0X5F;        // Must begin with 0x or 0X
$hex_2 = 0x4a3b;      // Upper and lower case allowed
```

How to assign floating-point values

Using normal notation

```
$float_1 = 3.5;    // Must contain a decimal point
$float_2 = -6.0;   // May be negative
$float_3 = .125;   // Same as 0.125
$float_4 = 1.;     // Same as 1.0
```

Using exponential notation

```
$exp_1 = 9.451e15;   // Expands to 9.451 × 1015
$exp_2 = 6.022e+23;  // Plus sign is optional
$exp_3 = 1.602e-19;  // Exponent may be negative
$exp_4 = 9.806e0;    // Exponent may be zero
$exp_5 = -1.759e11;  // Mantissa may be negative
$exp_6 = 3e9;        // Mantissa may be a whole number
```

Two functions for working with infinity

```
is_infinite($value)
is_finite($value)
```

Working with infinity

Getting an infinite value

```
$inf_x = INF;          // Positive infinity, case-sensitive
$inf_x = -INF;           // Negative infinity
$inf_x = 1e200 * 1e200;  // Result is INF
$inf_x = 1 + INF;        // Result is INF
$inf_x = 1 / INF;        // Result is 0
$inf_x = 1 / 0;          // Generates a warning
```

Testing for an infinite value

```
$result = 1e200 * 1e200;
if (is_infinite($result)) {
    echo('Result was out of range.');
} else {
    echo('Result is ' . $result);
}
```

URL for a list of all PHP math functions

```
http://www.php.net/manual/en/ref.math.php
```

Common mathematical functions

```
abs($value)
ceil($value)
floor($value)
max($n1, $n2[, $n3 ...])
min($n1, $n2[, $n3 ...])
pi()
pow($base, $exp)
round($value[, $precision])
sqrt($value)
```

How to round a number

```
$subtotal = 15.99;
$tax_rate = 0.08;
$tax = round($subtotal * $tax_rate, 2);
```

How to get the square root of a number

```
$num1 = 4;
$root  = sqrt($num1);
```

How to work with exponents

```
$num2 = 5;
$power = pow($num2, 2);
```

How to calculate the distance between two points

```
$x1 = 5; $y1 = 4;
$x2 = 2; $y2 = 8;
$distance = sqrt(pow($x1 - $x2, 2) + pow($y1 - $y2, 2));
```

How to place a maximum bound on a number

```
$value = 15;
$max_value = 10;
$value = min($max_value, $value);   // 10
```

Functions that generate random numbers

```
getrandmax()
rand()
rand($lo, $hi)
mt_getrandmax()
mt_rand()
mt_rand($lo, $hi)
```

How to simulate a random dice roll

```
$dice = mt_rand(1, 6);
```

How to generate a random value between 0 and 1 with 5 decimal places

```
$number = 0;
$places = 5;
for($i = 0; $i < $places; $i++) {
    $number += mt_rand(0,9);
    $number /= 10;
}
echo $number;
```

How to generate a random password

```
$password_length = 8;

// Add a symbol to the password
$symbols  = '~!@#$%^&*()-_=+[]{};:,.<>?';
$symbol_count = strlen($symbols);
$index = mt_rand(0, $symbol_count - 1);
$password = substr($symbols, $index , 1);

$password .= chr(mt_rand(48, 57));  //random numeric char
$password .= chr(mt_rand(65, 90));  //random upper char

// Add lowercase letters to reach the specified length
while (strlen($password) < $password_length) {
    $password .= chr(mt_rand(97, 122));
}

$password = str_shuffle($password);
echo $password;
```

The sprintf function

```
sprintf($format, $val1[, val2 ...])
```

Data type code

Character	Formats…
s	The value as a string.
d	The value as an integer.
f	The value as a floating-point number.
e	The value using exponential notation.
c	An integer value as its corresponding ASCII character.
b	An integer value as a binary number.
o	An integer value as an octal number.
x	An integer value as a hexadecimal number (lowercase).

A sprintf function that formats two values

```
$message = sprintf('The book about %s has %d pages.',
                   'PHP', 800);
```

How to use sprintf to convert numbers to strings

```
$s1 = sprintf('It cost %s dollars', 12);
$s2 = sprintf('%s', 4.5);
$s3 = sprintf('%s', 9451000.000000);
$s4 = sprintf('%f', 9.451e6);
$s5 = sprintf('%e', 9451000.000000);
$s6 = sprintf('%c', 65);
$s7 = sprintf('%x', 15);
$s8 = sprintf('%X', 15);
$s9 = sprintf('%s%%', 4.5);
```

Two functions for converting strings to numbers

```
intval($var)
floatval($var)
```

How to convert a string to an integer

Using type casting

```
$value_1 = (int) '42';
$value_2 = (int) '42.5';
$value_3 = (int) '42 miles';
$value_4 = (int) '2,500 feet';
$value_5 = (int) 'miles: 42';
$value_6 = (int) 'miles';
$value_7 = (int) '10000000000';
$value_8 = (int) '042';
$value_9 = (int) '0x42';
```

Using the intval function

```
$value = intval('42');
```

How to convert a string to a floating-point number

Using type casting

```
$value_1 = (float) '4.2';
$value_2 = (float) '4.2 gallons';
$value_3 = (float) 'gallons';
$value_4 = (float) '1.5e-3';
$value_5 = (float) '1e400';
```

Using the floatval function

```
$value = floatval('4.2');
```