PHP Advanced

4.1 PHP - Multidimensional Arrays

A multidimensional array is an array containing one or more arrays.

PHP understands multidimensional arrays that are two, three, four, five, or more levels deep. However, arrays more than three levels deep are hard to manage for most people.

The dimension of an array indicates the number of indices you need to select an element.

- For a two-dimensional array you need two indices to select an element
- For a three-dimensional array you need three indices to select an element

4.2 PHP - Two-dimensional Arrays

A two-dimensional array is an array of arrays (a three-dimensional array is an array of arrays of arrays).

First, take a look at the following table:

Name	Stock	Sold
Volvo	22	18
BMW	15	13
Saab	5	2
Land Rover	17	15

We can store the data from the table above in a two-dimensional array, like this:

```
$cars = array
  (
  array("Volvo",22,18),
  array("BMW",15,13),
  array("Saab",5,2),
```

```
  array("Land Rover",17,15)
  );
```

Now the two-dimensional $cars array contains four arrays, and it has two indices: row and column.

To get access to the elements of the $cars array we must point to the two indices (row and column):

Example:

```
<!DOCTYPE html>
<html>
<body>
<?php
$cars = array
  (
  array("Volvo",22,18),
  array("BMW",15,13),
  array("Saab",5,2),
  array("Land Rover",17,15)
  );
 echo $cars[0][0].": In stock: ".$cars[0][1].", sold: ".$cars[0][2].".<br>";
echo $cars[1][0].": In stock: ".$cars[1][1].", sold: ".$cars[1][2].".<br>";
echo $cars[2][0].": In stock: ".$cars[2][1].", sold: ".$cars[2][2].".<br>";
echo $cars[3][0].": In stock: ".$cars[3][1].", sold: ".$cars[3][2].".<br>";
?>
</body>
</html>
```

Output:

```
 Volvo: In stock: 22, sold: 18.
BMW: In stock: 15, sold: 13.
Saab: In stock: 5, sold: 2.
Land Rover: In stock: 17, sold: 15.
```

We can also put a for loop inside another for loop to get the elements of the $cars array (we still have to point to the two indices):

```
<!DOCTYPE html>
<html>
<body>

<?php
$cars = array
  (
  array("Volvo",22,18),
  array("BMW",15,13),
  array("Saab",5,2),
  array("Land Rover",17,15)
  );

for ($row = 0; $row < 4; $row++) {
  echo "<p><b>Row number $row</b></p>";
  echo "<ul>";
  for ($col = 0; $col < 3; $col++) {
    echo "<li>".$cars[$row][$col]."</li>";
  }
  echo "</ul>";
}
?>
</body>
</html>
```

Output:

Row number 0

- Volvo
- 22
- 18

Row number 1

- BMW
- 15
- 13

Row number 2

- Saab
- 5
- 2

Row number 3

- Land Rover
- 17
- 15

4.3 PHP 5 Date and Time

4.3.1 The PHP Date() Function

The PHP date() function formats a timestamp to a more readable date and time.

Syntax
date(*format*,*timestamp*)

Parameter	Description
format	Required. Specifies the format of the timestamp
timestamp	Optional. Specifies a timestamp. Default is the current date and time

A timestamp is a sequence of characters, denoting the date and/or time at which a certain event occurred.

4.4 Get a Simple Date

The required *format* parameter of the date() function specifies how to format the date (or time).

Here are some characters that are commonly used for dates:

- d - Represents the day of the month (01 to 31)
- m - Represents a month (01 to 12)
- Y - Represents a year (in four digits)

- l (lowercase 'L') - Represents the day of the week

Other characters, like"/", ".", or "-" can also be inserted between the characters to add additional formatting.

The example below formats today's date in three different ways:

Example:

```
<!DOCTYPE html>
<html>
<body>
<?php
echo "Today is " . date("Y/m/d") . "<br>";
echo "Today is " . date("Y.m.d") . "<br>";
echo "Today is " . date("Y-m-d") . "<br>";
echo "Today is " . date("l");
?>
</body>
</html>
```

Output:

Today is 2018/12/14
Today is 2018.12.14
Today is 2018-12-14
Today is Friday

4.5 PHP Tip - Automatic Copyright Year

Use the `date()` function to automatically update the copyright year on your website:

Example: © 2010-2018

4.5.1 Get a Simple Time

Here are some characters that are commonly used for times:

- h - 12-hour format of an hour with leading zeros (01 to 12)
- i - Minutes with leading zeros (00 to 59)
- s - Seconds with leading zeros (00 to 59)
- a - Lowercase Ante meridiem and Post meridiem (am or pm)

The example below outputs the current time in the specified format:

Example:

```
<!DOCTYPE html>
<html>
<body>
<?php
echo "The time is " . date("h:i:sa");
?>
</body>
</html>
```

Output: The time is 11:45:07am

Note that the PHP date() function will return the current date/time of the server!

4.5.2 Get Your Time Zone

If the time you got back from the code is not the right time, it's probably because your server is in another country or set up for a different timezone. So, if you need the time to be correct according to a specific location, you can set a timezone to use.

The example below sets the timezone to "America/New_York", then outputs the current time in the specified format:

Example:

```
<!DOCTYPE html>
<html>
<body>
<?php
date_default_timezone_set("America/New_York");
echo "The time is " . date("h:i:sa");
?>
</body>
</html>
```

Output: The time is 11:45:30am

4.5.3 Create a Date With PHP mktime()

The optional *timestamp* parameter in the date() function specifies a timestamp. If you do not specify a timestamp, the current date and time will be used (as shown in the examples above).The `mktime()` function returns the Unix timestamp for a date. The Unix timestamp contains the number of seconds between the Unix Epoch (January 1 1970 00:00:00 GMT) and the time specified. Syntax

mktime(*hour,minute,second,month,day,year*)

The example below creates a date and time from a number of parameters in the `mktime()` function: Example:

```
<!DOCTYPE html>
<html>
<body>
<?php
$d=mktime(11, 14, 54, 8, 12, 2014);
echo "Created date is " . date("Y-m-d h:i:sa", $d);
?>
```

</body>
</html>

Output:: Created date is 2014-08-12 11:14:54am

5.2.4 Create a Date From a String With PHP strtotime()

The PHP strtotime() function is used to convert a human readable string to a Unix time.

Syntax
strtotime(*time,now*)

The example below creates a date and time from the strtotime() function: Example:

```
<!DOCTYPE html>
<html>
<body>
<?php
$d=strtotime("10:30pm April 15 2014");
echo "Created date is " . date("Y-m-d h:i:sa", $d);
?>
</body>
</html>
```

Output: Created date is 2014-04-15 10:30:00pm

PHP is quite clever about converting a string to a date, so you can put in various values:

Example:

```
<!DOCTYPE html>
<html>
<body>
<?php
$d=strtotime("tomorrow");
echo date("Y-m-d h:i:sa", $d) . "<br>";
$d=strtotime("next Saturday");
echo date("Y-m-d h:i:sa", $d) . "<br>";
$d=strtotime("+3 Months");
echo date("Y-m-d h:i:sa", $d) . "<br>";
?>
</body>
</html>
```

Output: 2018-12-15 12:00:00am
2018-12-15 12:00:00am
2019-03-14 11:51:08am

However, strtotime()is not perfect, so remember to check the strings you put in there.

*More Date Examples

The example below outputs the dates for the next six Saturdays:

Example:

```
<!DOCTYPE html>
<html>
<body>
<?php
$startdate=strtotime("Saturday");
$enddate=strtotime("+6 weeks", $startdate);
while ($startdate < $enddate)

 {
  echo date("M d", $startdate) . "<br>";
  $startdate = strtotime("+1 week", $startdate);
}
?>
</body>
</html>
```

Output:

Dec 15
Dec 22
Dec 29
Jan 05
Jan 12
Jan 19