Chapter 3 Opening a File

In this chapter I want to talk about opening a file using PHP. It is more or less the same as opening up a Word document or Excel spreadsheet where you choose File, Open, click on the filename and choose Open and boom the file is displayed on the screen (of course you may have just double clicked the file name as a shortcut). Our job is to tell the computer to do the same thing. We are going to open a csv (comma separated values) file which is just like an Excel spreadsheet without any formatting. We are going to be using only csv files for the course though you can open other types of files.

To open a file we use the built in PHP function fopen [http://php.net/manual/en/function.fopen.php]. Here is the general syntax:

```
fopen('path/filename', 'mode')
```

The path /filename is the folder and what you called the file. The mode is how you want to open the file and the manual lists all the possibilities. For our purpose there are three that we use:

1. `'r'` Opens the file for reading only.
2. `'a'` Opens the file for writing, appending to the end of the file.
3. `'w'` Opens the file for writing, always starting fresh and deleting what was previously saved in the file.

For writing (both modes a, w) the path must exist (i.e. you need to make the folder) but the file does not have to exist. Both statements will create a new file if the file does not exist already.

An example of the fopen command in use would look like this:

```
$file=fopen('data/food.csv', 'r');
```

Notice we have a folder named **data** with the file name **food.csv**. We assign the variable $file to the result of the fopen statement which would be a connection from the PHP file to the file you have opened. If the connection cannot be made $file is assigned FALSE.

My csv file looks like this:

```
pmkId,fldUsername,fldDescription,fldCalories,fldLinkToFoodGawker,fldImageUrl
1,rerickso,Breakfast is served,189,http://foodgawker.com/post/2012/11/27/207904/,muffin_thumb.jpg
2,rerickso,Lunch is served,220,http://foodgawker.com/post/2012/11/27/207904/,muffin_thumb.jpg
```

In Excel it looks like this:

	A	B	C	D	E	F
1	pmkId	fldUsername	fldDescription	fldCalories	fldLinkToFoodGawker	fldImageUrl
2	1	rerickso	Breakfast is served	189	http://foodgawker.com/post/2012/11/27/207904/	muffin_thumb.jpg
3	2	rerickso	Lunch is served	220	http://foodgawker.com/post/2012/11/27/207904/	muffin_thumb.jpg
4						

One of the biggest mistakes everyone has is not actually having the file present on the server (your forgot to upload it) or having the filename incorrect. So to help us "debug" [https://en.wikipedia.org/wiki/Debugging#Origin] or find the mistakes in our computer code we can add a few variables. First create a 'flag' variable called debug which we can use to decide if we want to display a debugging value or not.

```
$debug = false;
```

When you want to see your debugging output you can set this to true. To make it easier for our classroom we can add a statement (never done live, only for a class) that allows us to pass in a value to the web page and set debug to true. I put this code at the beginning of the document.

```
if (isset($_GET['debug'])) {
      $debug = true;
}
```

What `isset` does is that it looks in the URL GET to see if there is a variable called debug which would come after the fielname and question mark. Remember this is only done for a classroom setting to make it easier for us to help you. The URL would look like this:

http://rerickso.w3.uvm.edu/cs008/tryme.php?debug=t

Where we just append the variable name and value debug=t to the end. We use a ? mark to separate the file name and the GET variable as they are called. Of course by itself this does nothing. We would have to add a few print statements to print out the values so we can see what is in the computer's memory. It is easier to put the filename in a variable so that we can do this. So instead of opening the filename we open the variable. This is the same thing only using a variable makes for more flexible code. Here I have separated the filename into three variables and printed it out.

```
$myFolder = 'data/';
$myFileName = 'food';
$myfileExt = '.csv';
$filename = $myFolder . $myFileName . $myfileExt;
if ($debug) print '<p>filename is ' . $filename;
```

It may look like I made a typo with my if statement as it is all on one line. However this is ok since we have only one statement. It is not always the best thing to do as if you want to add another line of code to the if statement you would have to add the curly braces.

```
if ($debug) print '<p>filename is ' . $filename;
```

Is the same as:

```
if ($debug) {
   print '<p>filename is ' . $filename;
}
```

However this:

```
if ($debug) print '<p>filename is ' . $filename;
            print '<p>Testing without {}';
```

Is NOT the same as:

```
if ($debug) {
   print '<p>filename is ' . $filename;
   print '<p>Testing with {}';
}
```

In the first if statement, "testing without {}" will always print because without the curly braces the if statement only applies to one line. The second if would only print "Testing with {}" if $debug was set to true. Clear as mud?

Printing out the path/filename is useful so that you can compare it to what you have setup on your computer and uploaded to the server. I also add a debug statement to let me know if the file was opened. Here is the whole block of code to open a file. The good news is that the only part that ever changes is the filename. The rest stays the same.

```php
<?php
// Open a CSV file
$debug = false;
if (isset($_GET["debug"])) {
     $debug = true;
}

$myFolder = 'data/';

$myFileName = 'food';

$fileExt = '.csv';

$filename = $myFolder . $myFileName . $fileExt;

if ($debug) print '<p>filename is ' . $filename;

$file=fopen($filename, "r");

if ($debug) {
    if ($file) {
       print '<p>File Opened Successfully.</p>';
    } else {
       print '<p>File Open Failed.</p>';
     }
} ?>
```

The above code block is something you save and use over and over again anytime you need to open a file. Technically the shortest way would be to have one line of code like this:

```php
<?php
// Open a CSV file
$file=fopen('data/food.csv', "r");
?>
```

better would be to use a variable but you can use just one like this:

```php
<?php
// Open a CSV file
$filename = 'data/food.csv';
$file = fopen($filename, "r");
?>
```

I added the rest of the code to help make it easier for me to help you when you make a mistake.

Now all the code above does is open a file so we now need to read the file into the computer's memory so we can do something with it. To read a csv file we use the PHP function fgetcsv which reads one line at a time. The syntax is where $file is the variable you opened the file with:

```
fgetcsv($file);
```

We always put the file into an array so we can access each row at a time. Since many csv files come with a header row (text describing the data in each column) I always read the header into its own array like this:

```
$headers[]=fgetcsv($file);
```

If there are more than one header rows just repeat the line twice. I am appending the row to an array so you can read as many times as you want. On the other hand, if you don't have a header row just delete that line. A couple of points to keep in mind:

1. Always make sure the file is opened with a simple if statement using the same file variable name (in my examples it has always been $file).
2. It is a good idea to add debug statements so you can see what is in your headers. If all your data is in your headers there is a problem with your csv file.

Next we need to read all the data into an array. (Name the array based on the data in the csv file. For example mine is about foods so that is what I will name my array, giving the name a plural name) We want to keep reading until there are no more records left so we need a repetition structure. Since we do not know how many times to read the file (it may be none) we are going to use a while loop [http://php.net/manual/en/control-structures.while.php] checking to see if we are at the end of the file (feof [http://php.net/manual/en/function.feof.php]). Each time in the loop we append to the end of our array what is returned by the fgetcsv statement that we used to read the headers. Here is the complete code block to read a file into an array:

```php
<?php
if ($file) {
    if ($debug) print '<p>Begin reading data into an array</p>';

    // read the header row, copy the line for each header row
    // you have.
    $headers[] = fgetcsv($file);

    if ($debug) {
         print '<p>Finished reading headers.</p>';
         print '<p>My header array:</p><pre>';
         print_r($headers);
         print '</pre>';
     }

     // read all the data
     while (!feof($file)) {
         $foods[] = fgetcsv($file);
     }

     if ($debug) {
         print '<p>Finished reading data. File closed.</p>';
         print '<p>My data array<p><pre> ';
         print_r($foods);
         print '</pre></p>';
     }
} // ends if file was opened
?>
```

What is nice about this code block is there are only two things you ever need to change when reading in a csv file:

1. If you don't have a header row you don't read the header row. If you have two lines of headers in the csv then you need to read two lines by coping the line `$headers[] = fgetcsv($file);`
2. Change the name of the array in two places. For example my array is called foods so if my data was about trees I would inside the while loop change foods to trees and then again in the debug if block.

We have opened our file and read the contents into an array in memory. We should close the file using the PHP function fclose [] which looks like this. Again the variable I used was $file.

```
fclose($file);
```

The last step is to do something with your data. This is also the hardest part as it is not always the same, unlike the previous code where you only needed to change one or two things. Since our data is in an array we will use a foreach loop to print it out. It is best to think of the HTML you want to create before trying to print it out. I am going to print out my foods in an ordered list but inside that ordered list I want to get a little fancy for styling reasons. Here is what I want it to look like (see sample csv at the beginning of the chapter as that is where the data (information) is coming from):

Of course the look is all CSS but here is the HTML markup I used inside the li element:

```
<a href="http://foodgawker.com/post/2012/11/27/207904/" target="_blank" >
    <img src="images/muffin_thumb.jpg" alt="Breakfast is served">
</a>
<span class="userId">rerickso</span>
<span class="description">Breakfast is served</span>
<span class="calories">189</span>
```

Ok the hard part is to notice the data is all going to be replaced by the array variable and column number. My array name is $foods so in a foreach loop the row would be called $food following our naming conventions. Looking at the first line, the first piece of data is the URL for the hyper link. Looking in our csv file / Excel file that is column E which is the fifth column but since computers start counting with zero it would $food[4]. So that first line would look like this in PHP:

```
print '<a href="' . $food[4] . '" target="_blank" ' . '>';
```

See how I put single quotes around the text that stays the same? Then I concatenate the variable for the data? In the end my foreach loop looks like this to display the data:

```
<?php
// display the data
print '<ol>';
foreach ($foods as $food) {
  if ($food[0] != "") {  //the eof would be a ""
     print '<li>';
     print '<a href="' . $food[4] . '" target="_blank" ' . '>';
     print '<img src="images/' . $food[5] . '" alt="' . $food[2]
. '">';
     print '</a>';
     print '<span class="userId">' . $food[1] . '</span>';
     print '<span class="description">' . $food[2] . '</span>';
     print '<span class="calories">' . $food[3] . '</span>';
     print '</li>';
  }
}
print '</ol>';

if ($debug) print '<p>End of processing.</p>';

?>
```

Notice I added an if statement inside the loop to make sure that I had data in the first column. This is because the end of the file shows up in my array. Secondly I added a debug statement to just print out I was at the end to indicate all my code had been executed.

Of course the look all comes from the CSS applied to the HTML and though not part of this chapter on files here is the CSS I used for the example:

```
img{
      border: .1em solid #555555;
      padding: .2em;
}

img:hover{
      background-color: #D39745;
}

ol{
      text-align: center;
}

ol li{
      background-color: #F3EFE0;
      border: thin solid darkgray;
      border-radius: 15px;
      box-shadow: 10px 10px 5px #888888;
      display: inline-block;
      list-style-type: none;
      margin: 1em;
      padding: 1em;
      text-align: center;
      width: 320px;
}

ol + li{
      margin-left: 0;
}

.userId{
      float: right;
      color: #028198;
```

```
}

.description{
     clear: both;
     color: #555555;
     display: block;
     text-align: left;
}

.calories{
     color: #555555;
     display: block;
     padding-top: 2em;
     text-align: left;
}
```

Self Test Questions

topics:

- opening a file (fopen)
- reading, appending, writing
- csv files
- debug
- one line if statements w/o brackets
- fgetcsv
- fclose

Challenge Questions

1. **Animals**
   Create a file named 'animals.csv' and fill it with your favorite animals, separated by a comma.
   Write some php code that displays all of the animals in the file.

2. **First Five Entries**
   Create a file 'file.csv' and fill it with whatever you'd like, delimited by commas. Write some php code that only displays the first five elements of the file.

3. **Random Items**
   Create a file 'file.csv' and fill it with whatever you'd like, delimited by commas. Make sure there are at least 10 or so entries. For this exercise we will use the php function 'rand' which takes two integer values $min and $max and chooses a random value between them, inclusive.

   rand(1,10) could output 5 for example.

   Using the *rand()* function, print a random item from your file on screen. For the minimum value, use 0. For the maximum value, enter in the number of items in your file (or get the size of the array using the count() function)

4. **Average of numbers**
   Create a file named 'numbers.csv' and fill it with as many **integers** as you want, delimited by commas. Write some php code that prints the average of all the numbers in the file.

   For this problem you will need the php function (int) which converts a string value into an integer value. It works as follows:

   $integer = (int) $string;

5. **Item Counter**

Create a file named ‘shoppinglist.csv’ and fill it with your shopping list, delimited by commas. Write some php code that displays the number of items in the shopping list. (Hint: use the php count() function)

Answers to Challenge Questions

1. Answer:

```
<?php
	$file = fopen(‘animals.csv’, “r”);
	while ($feof($file)){
		$animals[] = fgetcsv($file);
	}
	foreach($animals as $animal){
		print '<p>' . $animal . '</p>';
	}
	fclose($file);
?>
```

2. Answer:

```
<?php
	$file = fopen(“file.csv”, “r”);
	while($foef($file)){
		$stuff[] = fgetcsv($file);
	}

	print '<p>' . $stuff[0] . '</p>';
	print '<p>' . $stuff[1] . '</p>';
	print '<p>' . $stuff[2] . '</p>';
	print '<p>' . $stuff[3] . '</p>';
	print '<p>' . $stuff[4] . '</p>';
?>
```

You could use for loop for as well

3. Answer:

```
<?php
	$file = fopen(“file.csv”, “r”);
	while($foef($file)){
		$stuff[] = fgetcsv($file);
	}
	$randInt = $rand(0,count($stuff));

	print '<p>' . $stuff[$randInt] . '</p>';
?>
```

4. Answer:

```
<?php
        $file = fopen(“numbers.csv”, “r”);
        while($foef($file)){
                $numbers[] = fgetcsv($file);
        }
        $sum = 0;
        foreach($numbers as $number){
                $sum += (int) $number;
        }
        $average = $sum / count($numbers);
        print '<p>' . $average . '</p>';
?>
```

5. Answer:

```
<?php
        $file = fopen(“shoppinglist.csv”, “r”);
        while($foef($file)){
                $shoppingList[] = fgetcsv($file);
        }
        $numItems = count($shoppingList);
        print '<p>' . $ numItems. '</p>';

?>
```