Chapter 4 Functions

We have been using PHP built in constants like PHP_EOL for a \n to print the end of line in our output. We have also used the built in PHP function `fopen('path/filename', 'mode')`. That has two parameters that it needs. A new built in variable in PHP that we are going to use is:

```
$_SERVER['PHP_SELF']
```

Which you may notice is an array (because of the square brackets). PHP_SELF is just one of the variables in the array which holds the information after the domain name. So for example if my url is:

https://rerickso.w3.uvm.edu/cs008/tryme.php

then PHP_SELF would have:

cs008/tryme.php

This is useful to use for a few reasons, I will explain one. I can use a built in PHP function called `pathinfo` which will separate cs008/tryme.php into the different parts (note I will explain htmlentities in a minute.

```
$phpSelf = htmlentities($_SERVER['PHP_SELF'], ENT_QUOTES, "UTF-8");
// break the url up into an array, then pull out just the filename
$path_parts = pathinfo($phpSelf);
```

Now path parts is an array we can pull out just the name of the file like this:

```
$path_parts['filename']
```

which would just have the filename tryme. The reason I would like to do this is because every html body element is supposed to have an id equal to the file name. We can have php do this automatically like this:

```
print '<body id="' . $path_parts['filename'] . '">';
```

Of course we need all the code like this:

```
<?php
// parse the url into htmlentities to remove any suspicious vales that
// someone may try to pass in. htmlentities helps avoid security issues.
$phpSelf = htmlentities($_SERVER['PHP_SELF'], ENT_QUOTES, "UTF-8");

// break the url up into an array, then pull out just the filename
$path_parts = pathinfo($phpSelf);
?>
```

```
<!DOCTYPE html>
<html lang="en">
     <head
         <title>read this http://moz.com/learn/seo/title-tag </title>
         <meta charset="utf-8">
         <meta name="author" content="Robert M. Erickson">
         <meta name="description" content="read this:
http://moz.com/learn/seo/meta-description ">
          <!-- see: http://webdesign.tutsplus.com/tutorials/htmlcss-
tutorials/quick-tip-dont-forget-the-viewport-meta-tag/ -->
         <meta name="viewport" content="width=device-width, initial-scale=1">
         <link rel="stylesheet" href="style.css" type="text/css"
media="screen">
      </head>

<?php // giving each body tag an id really helps with css later on
print '<body id="' . $path_parts['filename'] . '">';
?>
```

There is a built-in function that we will use all the time called `htmlentities` [http://php.net/manual/en/function.htmlentities.php] and it's purpose is to help with security of our web sites by taking potential malicious information and converting it to a harmless html entity [https://www.w3schools.com/html/html_entities.asp] . An html entity is how I display the html code on the web page. For example the less than symbol is always the start of an html element so in order to display a less than symbol we use the html entity in our code $lt; which the browser will display as < (less than symbol).

Now let's learn how to make our own functions. Functions "Encapsulate" (enclose) a task that you are coding. What this does for us is to have a one line **function call** that can replace many lines of PHP code in our program. Making the code shorter is almost always a good thing.

Ok first maybe we should talk about why you make a function. Generally if you find yourself writing a piece of code once and then coping and pasting that code to another location you should start to think to yourself should I make a function for this? Just to let you know you do get better at this as you go along, it comes with experience and ten years of PHP programming you will start to have a really good handle on what works and what does not work well.

On to functions. A function is created with a function declaration statement like this (NOTE: there will be a closing } at the end of the function):

```
function DescriptiveName([$variable names for inputs needed]){
```

Where the blue word function is a PHP reserved word to identify this as a function. The green text is the name you give to your function. I generally follow CamelCase notation with

the first letter capitalized unlike variables where the first letter is lowercase. Inside the parentheses we have the variables that the function needs. You will notice the [square brackets] which identify that this is optional (you never actually type the square brackets, they are for an array). Not all functions require a parameter and some require several. A programmer tries to minimize the number of parameters a function needs (more than three gets bulky, but its never a perfect world and you may see functions with more than three).

Generally I place my functions in a separate file (that contains many functions, though sometimes only one) that I include like other files when I need them. Let's have an example where we want to make a function that will calculate the area of a Bedroom (or kitchen, living room, deck or in math terms a square or rectangle). There is not a lot of math in this course but we should be able to take a mathematical formula and code it into a function. The area is calculated as the length times the width. Let us look at this studio apartment:

The area of the studio space would be 13 times 19 equals 247 square feet. Let us write the function to return this calculation. We could write the function to print the value but most times it is better to return the value as in some cases we may not want to print it right away (Maybe you want to add up all the square feet for each room). We start off by thinking of a good name for our function, hmm how about calculate area (CalculateArea in camel case). What values do we need to do this? We need length and width. Here is what our function declaration statement would look like:

```
function CalculateArea($length, $width){
```

Since we want to get the area I will make a variable with that name and initialize it out of habit. I also add some comments to clear things up. The whole function would like this:

```
/* given the inputs this function returns the area
  $length should be greater than zero
  $width should be greater than zero
*/
function CalculateArea($length, $width){
      $area = "";
      $area = $length * $width;
      return $area;
}
```

And the function call would look like this:

```
print "<p>The Apartment is: " . CalculateArea(19,13) . " square
feet.</p>";
```

Of course I have just hard coded the values. Normally you would most likely have the person fill out a form. A better way to code this would be to use variables like this:

```
<?php
$length = 19;
$width = 13;
print "<p>The Apartment is: " . CalculateArea($length,$width) . " square
feet.</p>";
```

The next step would be to pass the values into the web page using the GET format until you learn how to make a form. So what do you think of this code:

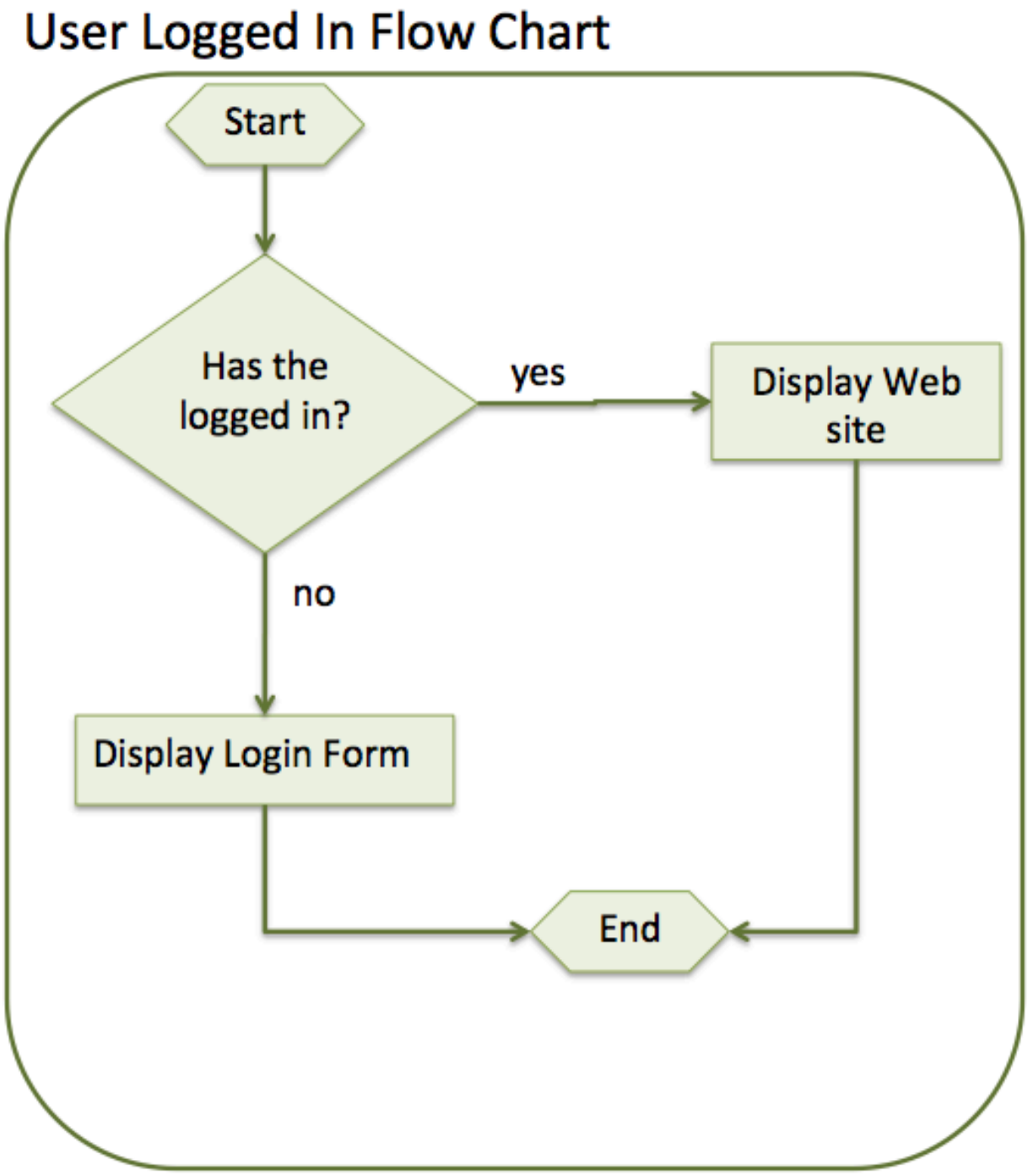
```
<?php
/* given the inputs this function returns the area   $length should be
greater than zero     $width should be greater than zero    */
function CalculateArea($length, $width){
      $area = "";
      $area = $length * $width;
      return $area;
}

$length = (int) htmlentities($_GET["length"], ENT_QUOTES, "UTF-8");
$width = (int) htmlentities($_GET["width"], ENT_QUOTES, "UTF-8");
print "<p>The Apartment is: " . CalculateArea($length,$width) . " square
feet.</p>";
```

If we named this page square-feet.php and used the url to pass the values in the url would look like this:

https://rerickso.w3.uvm.edu/cs008/square-feet.php?length=19&width=13

All of the examples would print the same thing. Clear as mud? I want to introduce something called a dummy function that does not really do anything but act as a place holder until the code can be completed.

For example let's say you want to have someone log into your web site. However you don't know how to do that. Well you can pretend or hard code it for now until you learn or hire an outside contractor. So **logically** you only want to display the page if the person is logged in otherwise you want to display the login screen. A flow chart would look like this:



To check if a user is logged we would have to compare their username and password to what is on file etc.. However we can just code this with a dummy function so that we can continue working. You can have a function call in an if statement. Let's start with the function where we create the function declaration statement and return true to say yes the user is logged in. We would of course want to test this by returning false as well. Here is the function:

```
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// Check to see if user is logged in
function UserLoggedIn($username) {
     return true;
}
```

Our code in the file would look like this:

```
if (UserLoggedIn($username)) {
       // display page etc.
}else{
       // display login page. You can use an include
       // display page etc.
}
```

Since our function returns true we can just keep working on the page until we figure out how to do that. It is also a great way to test your logic in your page before spending time on writing a lot of code in a function.

So functions help us to consolidate our code. We can do something similar with our html code as well. Looking at the major boxes of our html they look like this where I have divided the page into six main sections:

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>read this http://moz.com/learn/seo/title-tag </title>
        <meta charset="utf-8">
        <meta name="author" content="Robert M. Erickson">
        <meta name="description" content="read this: http://moz.com/learn/seo/meta-description ">

        <!-- see: http://webdesign.tutsplus.com/tutorials/htmlcss-tutorials/quick-tip-dont-forget-the-viewport-meta-tag/ -->
        <meta name="viewport" content="width=device-width, initial-scale=1">

        <!--[if lt IE 9]>
                <script src="//html5shim.googlecode.com/svn/trunk/html5.js"></script>
        <![endif]-->

        <link rel="stylesheet" href="style.css" type="text/css" media="screen">



    </head>

    <body id="next">
    <!-- ######################     Start of Body   ########################## -->


<header>
    <h1>main title for SITE</h1>
</header>

<!-- ######################     Main Navigation   ########################## -->
<nav>
    <ol>
        <li><a href="index.php">Home</a></li><li class=" activePage "><a href="next.php">Next</a></li>    </ol>
</nav>

<article id="main">
    <h1>Next</h1>
    <p>Content Removed for shorter example.</p>
</article>

<aside id="other">
    <h2>Extra info</h2>
    <p>Content Removed again.</p>

</aside>

<!-- ######################     Footer   ########################################## -->
<footer>
        <p>Robert Erickson</p>
</footer>

</body>
</html>
```

The first section I call top because well it is just at the top. The top section includes the document type element to the opening body element. I take this code and I separate it not into a function but into a file. I name this file top.php. What php allows us to do is to include [see: http://php.net/manual/en/function.include.php] a file in another file. Conceptually it is having php open the file read all the content, copy it and then paste it where you put the include statement. So for example instead of typing the doc type in every page I type that section in a file. I then include that file. It would look something like this:

```
<?php
include "top.php";
```

There is nothing before this line not even white space.

The second section holds the header element where your companies name and logo go. I call this section the header section (ok maybe names aren't fancy but they do describe what they are) and I name the file header.php. I would then add to the code the include statement for the header so it looks like this:

```php
<?php
include "top.php";
include "header.php";
```

After the logo the next item that shows is the navigation section, which I name that file nav.php (I am getting lazy in my typing so I abbreviated navigation). Adding the nav to our php we now have:

```php
<?php
include "top.php";
include "header.php";
include "nav.php";
?>
```

Sometimes I want my navigation html box inside my html head box so I just put the include nav statement inside the file header.php so it would look like this:

```
<!-- ####################     Page header    ############################ -->
<header>
     <h1>main title for SITE</h1>
</header>
<?php
include "nav.php";
?>
```

And of course I would NOT include nav.php in the my file.

Now the fourth section is the main body of your content and I have placed that in an article element (or a section element). This section is just all your content that changes from page to page. The first three sections don't really change from page to page.

The fifth section is what shows up at the bottom of the page so I call this the footer section, filename of footer.php of course. The code looks like this:

```php
<?php include ("footer.php"); ?>
```

I have placed the sixth section just in the page itself or in the footer.php, either way works. Here is how the code that you put in your file would look like:

```php
<?php
include ("top.php");
include ("header.php");
include ("nav.php");
?>
<article id="main">
    <h1>Next</h1>
    <p>(Show this <a href="next-source.php">page source</a>) Mama mihi adulescentem cum venerit sedeo me unicum filium auscultet quae loquor. Et si hoc proderit tibi futurum apricis die. O vestras tempore non vivunt nimias celeritates, mala venient et transibunt. Invenies mulier, scies amor menta filium quis est sursum. Et simplex qualem quaeso aliquid diligis, et intelligite. Baby sit unica qualem o non feceris mihi fili, si potes. Libido oblivisci divitis aurum totum quod est opus in anima tua. Et possis, o Infans sin omnibus cupio tibi, fili, sit satisfactum.</p>

    <p>Example of images in a table.</p>

    <table>
        <tr>
            <td><a href="pond-350.JPG" target="_blank"><img alt="pond-150 (56K)" src="pond-150.JPG"></a></td>
            <td><a href="pond-closeup-350.JPG" target="_blank"><img alt="pond-150 (56K)" src="pond-150.JPG"></a></td>
            <td></td>
        </tr>
    </table>

    <p>Example of images in a list. Notice the difference? Try resizing the screen.</p>

    <ol class="photos">
        <li><a href="pond-350.JPG" target="_blank"><img alt="pond-150 (56K)" src="pond-150.JPG"></a></li>
        <li><a href="pond-closeup-350.JPG" target="_blank"><img alt="pond-150 (56K)" src="pond-150.JPG"></a></li>
    </ol>


    <p>Et simplex qualem o aliquid diligis, et intelligite. Baby sit unica qualem o non feceris mihi fili, si potes. HEJA ego. Puer, ne solliciti estis, scies corde sequere et nihil aliud. Et possis, o Infans ll sin cupio tibi, fili, sit satisfactum. Et simplex qualem o aliquid diligis, et intelligite. Baby sit unica qualem O uos istud agas fili, si potes. Baby sit simplex, innocens vere o aliquid diligis, et intelligite.</p>
</article>

<aside id="other">
    <h2>Extra info</h2>
    <p>Mama mihi adulescentem cum venerit sedeo me unicum filium auscultet quae loquor. Et si hoc proderit tibi futurum apricis die. O vestras tempore non vivunt nimias celeritates, mala venient et transibunt. Invenies mulier, scies amor menta filium quis est sursum. Et simplex qualem quaeso aliquid diligis, et intelligite. Baby sit unica qualem o non feceris mihi fili, si potes. Libido oblivisci divitis aurum totum quod est opus in anima tua. Et possis, o Infans sin omnibus cupio tibi, fili, sit satisfactum.</p>

    <p>Et simplex qualem o aliquid diligis, et intelligite. Baby sit unica qualem o non feceris mihi fili, si potes. HEJA ego. Puer, ne solliciti estis, scies corde sequere et nihil aliud. Et possis, o Infans ll sin cupio tibi, fili, sit satisfactum. Et simplex qualem o aliquid diligis, et intelligite. Baby sit unica qualem O uos istud agas fili, si potes. Baby sit simplex, innocens vere o aliquid diligis, et intelligite.</p>
</aside>

<?php include ("footer.php"); ?>

</body>
</html>
```

Think about how a large web site with a thousand individual pages would work when you wanted to change one menu item. You would need to edit a thousand files. Now think about the same site when you use include files. You only need to change one file, nav.php and the whole thousand web pages are updated. The include statement is a pretty handy piece of code.

Summary

Functions make reusing code easier.

PHP comes with many built in functions and variables.

The function declaration tells you the name of the function and what parameters (values) the function needs.

Generally functions return a value.

Includes allow you to reuse your html (or php) code on more than one page.

Self Test Questions

1. Create a function to calculate a tip for your dinner bill. The user would need to enter the sub total and the % amount they want to tip. Have the function return the dollar amount to tip. You can pass the amounts into the web page using the GET format.
2. Create a function to calculate your tax on your dinner bill using a 9% meals tax . The user would need to enter the sub total. Have the function return the dollar amount of the tax. You can pass the amounts into the web page using the GET format.
3. Create a function that calls the previous functions to calculate your total bill. The user would need to enter the subtotal and % amount they want to tip. Use a 9% meals tax. Have the function return the total amount of the bill. You can pass the amounts into the web page using the GET format.
4. Put your functions above in a separate file called functions.php and include this file in top.php.
5. Edit your top.php so that it displays the name of the file as part of the title element.
6. **Kilometer Converter**

   Write some php code that converts a kilometer distance stored in a variable to miles. You should create a function to do the conversion. The conversion formula is as follows:

   Miles = Kilometers X 0.6214

7. **Automobile Costs**

   Write some php code that takes the monthly costs for the following expenses (stored in variables) incurred from operating an automobile: loan payment, insurance, gas, oil, tires, and maintenance. The page should display the total monthly cost of these expenses, and the total annual cost of these expenses. Be sure to create functions to solve this problem. (Optional addition: try loading these values from a csv file!)

8. **Stadium Seating**

   There are three seating categories at a stadium. For a softball game, Class A seats cost $20, Class B seats cost $15, and Class C seats cost $10. Write a php function that takes as input the number of tickets for each class of seats that were sold, and then displays on screen the amount of income generated from ticket sales.

9. **Paint Job Estimator**

   A painting company has determined that for every 112 square feet of wall space, one gallon of paint and eight hours of labor will be required. The company charges $35.00 per hour for labor. Write a php function that takes in the value of the square feet of wall space to be painted and the price of the paint per gallon. Then display on the page the following data:

   - The number of gallons of paint required.

- The hours of labor required
- The cost of the paint
- The labor charges
- The total cost of the paint job

10. **Maximum of Four Values**

    Write a function named *max* that accepts four integer values as arguments and returns the value that is the greater of the two. For example, if 7 , 3, 5, and 12 are passed as arguments to the function, the function should return 12. Print the returned number on screen.

Answers

Kilometer Converter Answer:

```
<?php

$kilometers = 50;

function Convert($kilometers){

        $kilometers = $miles * 0.6214;

        return $kilometers;

}

print "<p>" . $kilometers . " Kilometers is " . Convert($kilometers) . " Miles.</p>"

?>
```

Automobile Costs Answer:

```
<?php

$loan = 100

$insurance = 150

$gas = 100

$oil = 10

$tires = 50
```

```
$maintenance = 50


function totalMonthlyCost($l, $i, $g, $o, $t, $m){
        return $l + $i + $g + $o + $t + $m;
}
function totalAnnualCost($l, $i, $g, $o, $t, $m){
        return 12 * ($l + $i + $g + $o + $t + $m);
}


print “<p>Total monthly cost:” . totalMonthlyCost($loan, $insurance, $gas, $oil, $tires, $maintenance) . “</p>”;
print  “<p>Total annual cost:” . totalAnnualCost($loan, $insurance, $gas, $oil, $tires, $maintenance) . “</p>”;
?>
```

Stadium Seating Answer:

```
<?php
$numClassA = 147;
$numClassB = 462;
$numClassC = 1054;


function incomeGenerated($a, $b, $c){
        $total = $a * 20 + $b * 15 + $c * 10;
        return $total;
}
print “<p>The total amount of income generated was $” . incomeGenerated($numClassA,$numClassB,$numClassC) . “</p>;
```

```
?>
```

Paint Job Estimator Answer:

```
<?php
$wallSpace = 1000;
$paintPricePerGallon = 10;

function paintJob($ws, $pp){
        $numGallons = $ws / 112;
        $numHoursLabor = $numGallons * 8;
        $costOfPaint = $numGallons * $pp;
        $laborCharges = $numHoursLabor * 35;
        $totalJobCost = $costOfPaint + $laborCharges;

        print "<p>Number of gallons of paint required:" . $numGallons . "</p>";
        print "<p>Hours of labor required:" . $numHoursLabor . "</p>";
        print "<p>Paint cost:" . $costOfPaint . "</p>";
        print "<p>Labor charges:" . $laborCharges . "</p>";
        print "<p>Total cost of the paint job:" . $totalJobCost . "</p>";
}

paintJob($wallSpace,$paintPricePerGallon);
?>
```

Maximum of Four Values Answer:

```
<?php
function max($one,$two,$three,$four){
```

```
        $value = 0;
        if($one > $value){ $value = $one;}
        if($two > $value){ $value = $two;}
        if($three > $value){ $value = $three;}
        if($four > $value){ $value = $four;}
        return $value;
}

print max(7,3,5,12);
?>
```