Introduction to Web Programming

Lecture 8: Functions; File I/O

8.1: Functions

Functions

```
function name(parameterName, ..., parameterName) {
  statements;
}
```

```
function bmi($weight, $height) {
  $result = 703 * $weight / $height / $height;
  return $result;
}
```

- parameter types and return types are not written
- a function with no `return` statements is implicitly "void"
- can be declared in any PHP block, at start/end/middle of code

Calling functions

```
name(expression, ..., expression);
```

```
$w = 163;  # pounds
$h = 70;   # inches
$my_bmi = bmi($w, $h);
```

- if the wrong number of parameters are passed, it's an error

Variable scope: global and local vars

```
$school = "UW";                    # global
...

function downgrade() {
  global $school;
  $suffix = "(Wisconsin)";         # local

  $school = "$school $suffix";
  print "$school\n";
}
```

- variables declared in a function are **local** to that function; others are **global**
- if a function wants to use a global variable, it must have a `global` statement
  - but don't abuse this; mostly you should use parameters

Default parameter values

```
function name(parameterName = value, ..., parameterName = value) {
  statements;
}
```

```
function print_separated($str, $separator = ", ") {
  if (strlen($str) > 0) {
    print $str[0];
    for ($i = 1; $i < strlen($str); $i++) {
      print $separator . $str[$i];
    }
  }
}
```

```
print_separated("hello");        # h, e, l, l, o
print_separated("hello", "-");   # h-e-l-l-o
```

- if no value is passed, the default will be used (defaults must come last)

8.2: Advanced PHP Syntax

PHP file I/O functions

function name(s)	category
`file`, `file_get_contents`, `file_put_contents`	reading/writing entire files
`basename`, `file_exists`, `filesize`, `fileperms`, `filemtime`, `is_dir`, `is_readable`, `is_writable`, `disk_free_space`	asking for information
`copy`, `rename`, `unlink`, `chmod`, `chgrp`, `chown`, `mkdir`, `rmdir`	manipulating files and directories
`glob`, `scandir`	reading directories

Reading/writing files

contents of foo.txt	file("foo.txt")	file_get_contents("foo.txt")

Hello
how r u?
 	
I'm fine
	array(
	  "Hello\n", # 0
	  "how r u?\n", # 1
	  "\n", # 2
	  "I'm fine\n" # 3
	)
		"Hello\n
		how r u?\n # a single
		\n # string
		I'm fine\n"

- `file` function returns lines of a file as an array (`\n` at end of each)
- `file_get_contents` returns entire contents of a file as a single string
  - `file_put_contents` writes a string into a file

The `file` function

```
# display lines of file as a bulleted list
$lines = file("todolist.txt");
foreach ($lines as $line) {        # for ($i = 0; $i < count($lines); $i++)
  print $line;
}
```

- `file` returns the lines of a file as an array of strings
- each ends with `\n` ; to strip it, use an optional second parameter:

```
$lines = file("todolist.txt", FILE_IGNORE_NEW_LINES);
```

- common idiom: `foreach` or `for` loop over lines of file

Splitting/joining strings

```
$array = explode(delimiter, string);
$string = implode(delimiter, array);
```

```
$s  = "TIEI IWP 2018";
$a  = explode(" ", $s);     # ("TIEI", "IWP", "2018")
$s2 = implode("...", $a);   # "TIEI...IWP...2018"
```

- `explode` and `implode` convert between strings and arrays
- for more complex string splitting, you can use **regular expressions** (later)

Example with `explode`

contents of input file `names.txt`

```
Marc Frenchie Gaetano
Albert Genius Einstein
Tony IronMan Stark
```

```
foreach (file("names.txt") as $name) {
  $tokens = explode(" ", $name);
  ?>
  <p> author: <?= $tokens[2] ?>, <?= $tokens[0] ?> </p>
  <?php
}
```

```
author: Gaetano, Marc

author: Einstein, Albert

author: Stark, Tony
```

Unpacking an array: `list`

```
list($var1, ..., $varN) = array;
```

contents of input file `personal.txt`

```
Tony Stark
(206) 685 2181
570-86-7326
```

```
list($name, $phone, $ssn) = file("personal.txt");
...
list($area_code, $prefix, $suffix) = explode(" ", $phone);
```

- the odd `list` function "unpacks" an array into a set of variables you declare
- when you know a file or line's exact length/format, use `file` and `list` to unpack it

Reading directories

function	description
`glob`	returns an array of all file names that match a given pattern (returns a file path and name, such as `"foo/bar/myfile.txt"`)
`scandir`	returns an array of all file names in a given directory (returns just the file names, such as `"myfile.txt"`)

- `glob` can accept a general path with the `*` wildcard character (more powerful)

glob example

```
# reverse all poems in the poetry directory
$poems = glob("poetry/poem*.dat");
foreach ($poems as $poemfile) {
  $text = file_get_contents($poemfile);
  file_put_contents($poemfile, strrev($text));
  print "I just reversed " . basename($poemfile) . "\n";
}
```

- glob can match a "wildcard" path with the * character
  - glob("foo/bar/*.doc") returns all .doc files in the foo/bar subdirectory
  - glob("food*") returns all files whose names begin with "food"
- the basename function strips any leading directory from a file path
  - basename("foo/bar/baz.txt") returns "baz.txt"

scandir example

```
<ul>
  <?php foreach (scandir("taxes/old") as $filename) { ?>
    <li>I found a file: <?= $filename ?></li>
  <?php } ?>
</ul>
```

- .
- ..
- 2007_w2.pdf
- 2006_1099.doc

- scandir includes current directory (".") and parent ("..") in the array
- don't need basename with scandir; returns file names only without directory

Reading/writing an entire file

```
# reverse a file
$text = file_get_contents("poem.txt");
$text = strrev($text);
file_put_contents("poem.txt", $text);
```

- `file_get_contents` returns entire contents of a file as a string
  - if the file doesn't exist, you will get a warning and an empty return string
- `file_put_contents` writes a string into a file, replacing its old contents
  - if the file doesn't exist, it will be created

Appending to a file

```
# add a line to a file
$new_text = "P.S. ILY, GTG TTYL!~";
file_put_contents("poem.txt", $new_text, FILE_APPEND);
```

old contents	new contents
Roses are red, Violets are blue. All my base, Are belong to you.	Roses are red, Violets are blue. All my base, Are belong to you. P.S. ILY, GTG TTYL!~

- `file_put_contents` can be called with an optional third parameter to append (add to the end) rather than overwrite