SWINBURNE UNIVERSITY OF TECHNOLOGY

Advanced Web Development: Files and Directories

Week 5

Outline

- Handling String Input
- Managing files and directories
  - Comparing windows & Unix/Linux files and directories
- Working with Files
  - Opening and closing files
  - Writing data to files
  - Reading data from files

- Reading: Textbook Chapter 5
  PHP File Processing / File Streams:
  http://www.php.net/manual/en/refs.fileprocess.file.php

HANDLING STRING INPUT

`addslashes()` Function

- Accepts a single argument representing the text string you want to escape and returns a string containing the escaped string

```
$nickname = addslashes($_GET["nickname"]);
echo $nickname; // My best friend\'s nickname is \"Bubba\"
```

- Characters before backslash is added
  - `Single quote '`
  - `Double quote "`
  - `Backslash \`
  - `NULL`

`stripslashes()` Function

- Removes slashes that were added with the `addslashes()` function
- To prevent the display of escaped characters, use the `stripslashes()` function with the text you want to print

```
$nickname = stripslashes($_GET['nickname']);
echo $nickname; // My best friend's nickname is "Bubba"
```

MANAGING FILES AND DIRECTORIES

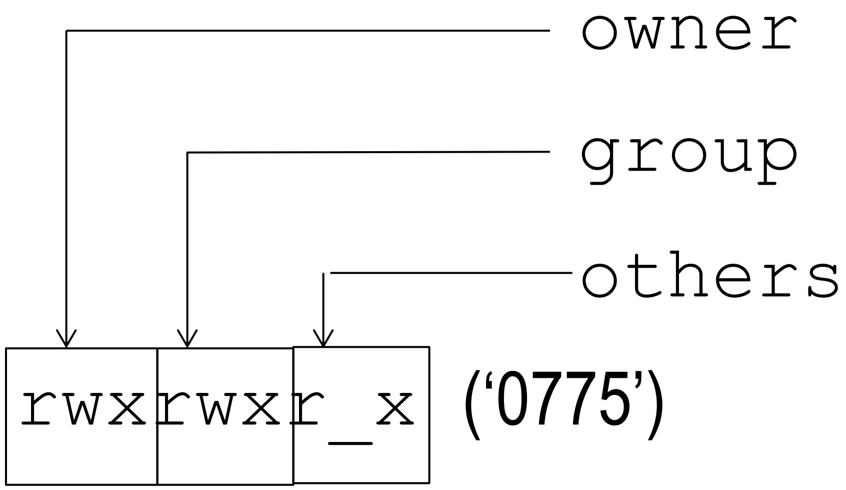
Windows & Unix/Linux File and Directory

- File is used to store data permanently for retrieval later
  - May used different end of line '\r' '\n' characters
- Directory a directory, also referred to as a folder is a virtual container within an electronic file system
  - "Directory" in Unix/Linux
  - "Folder" in Windows

Windows & Unix/Linux File and Directory

- Path delimiting character
  - ☐ Windows uses ‘\’, e.g. ‘cos30020\assign1’
  - ☐ Unix/Linux uses ‘/’, e.g. ‘cos30020/assign1’

- Unix/Linux has access permissions for directories/files e.g.



(‘0775’)

Reading Directories

PHP directory functions

Function	Description
chdir(directory)	Changes to the specified directory
chroot(directory)	Changes to the root directory
closedir($handle)	Closes a directory handle
getcwd()	Gets the current working directory
opendir(directory)	Opens a handle to the specified directory
readdir($handle)	Reads a file or directory name from a specified directory handle
rewinddir($handle)	Resets the directory pointer to the beginning of the directory
Scandir(directory[, sort])	Returns an indexed array containing the names of files and directories in the specified directory

Reading Directories (continued)

- To iterate through the entries in a directory, open a handle to the directory with the **`opendir()`** function
- Use the **`readdir()`** function to return the file and directory names from the open directory
- Use the **`closedir()`** function to close a directory handle

```
$dir = "C:\\PHP ";    // Windows path
$dirOpen = opendir($dir);
while ($curFile = readdir($dirOpen)) {
   echo $curFile , "<br />";
}
closedir($dirOpen);
```

`scandir()` Function

- Returns an indexed array containing the names of files and directories in the specified directory

```
$dir = "C:\\PHP"; // Windows path
$dirEntries = scandir($dir);
foreach ($dirEntries as $entry) {
        echo $entry , "<br />";
}
```

Creating Directories

- The **`mkdir()`** function creates a new directory
- To create a new directory within the current directory:
  - Pass just the name of the directory you want to create to the `mkdir()` function (on Windows)

```
mkdir("bowlers"); // how about on Unix?
```

- To create a new directory in a location other than the current directory:
  - Use a relative or an absolute path

```
mkdir("..\\tournament"); // Windows path
mkdir("C:\\PHP\\utilities");
```

  - Receive an error if exists

Obtaining File and Directory Information

PHP file and directory status functions

Function	Description
file_exists(filename)	Determines whether a file or directory exists
is_dir(filename)	Determine whether a filename is a directory
is_executable(filename)	Determines whether a file is executable
is_file(filename)	Determines whether a file is a regular file
is_readable(filename)	Determines whether a file is readable
is_writable(filename)	Determines whether a file is writable

Obtaining File and Directory Information (continued)

```
$dailyForecast = "<p><strong>San Francisco daily weather
   forecast</strong>: Today: Partly cloudy. Highs from the
   60s to mid 70s. West winds 5 to 15 mph. Tonight:
   Increasing clouds. Lows in the mid 40s to lower 50s.
   West winds 5 to 10 mph.</p>";
$weatherFile = "sfweather.txt";
if (is_writable($weatherFile)) {
    file_put_contents($weatherFile, $dailyForecast);
    echo "<p>The forecast information has been saved to
          the $weatherFile file.</p>";
} else {
    echo "<p>The forecast information cannot be saved to
          the $weatherFile file.</p>";
}
```

Obtaining File and Directory Information

(continued)

Function	Description
fileatime(filename)	Returns the last time a file was accessed
filectime(filename)	Returns the last time when the file was modified
fileowner(filename)	Returns the name of the file's owner
filetype(filename)	Returns the name of the file's owner
filesize(filename)	Returns the size of the file in bytes

Obtaining File and Directory Information

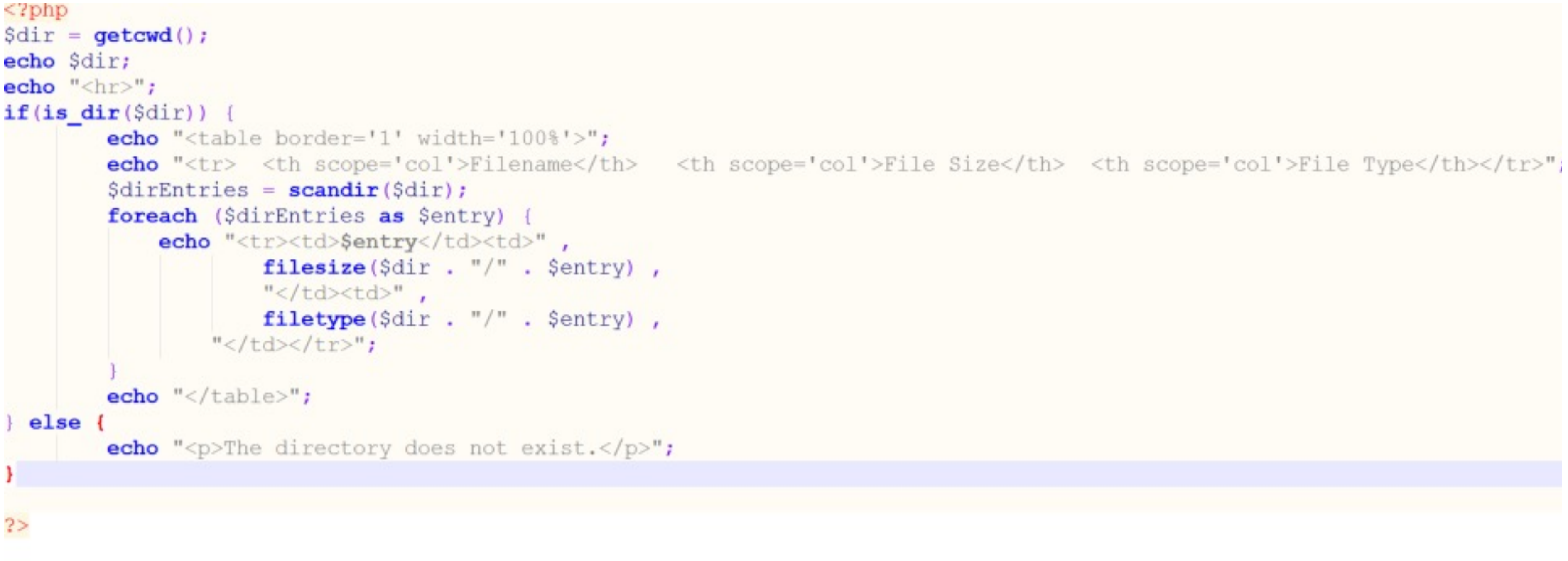
(continued)

```
$dir = "C:\\PHP"; // Windows path
if(is_dir($dir)) {
     echo "<table border='1' width='100%'>";
     echo "<tr>
           <th scope='col'>Filename</th>
           <th scope='col'>File Size</th>
           <th scope='col'>File Type</th></tr>";
     $dirEntries = scandir($dir);
     foreach ($dirEntries as $entry) {
           echo "<tr><td>$entry</td><td>" ,
                       filesize($dir . "\\" . $entry) ,
                       "</td><td>" ,
                       filetype($dir . "\\" . $entry) ,
                  "</td></tr>";
     }
     echo "</table>";
} else {
     echo "<p>The directory does not exist.</p>";
}
```

Note when to use , to separate items or . to concatenate

Obtaining File and Directory Information

(continued)

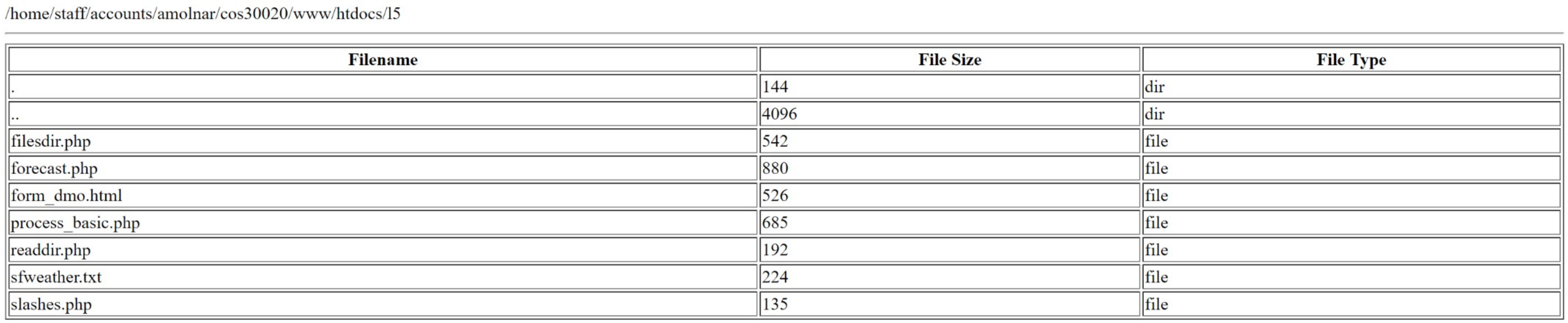


```
<?php
$dir = getcwd();
echo $dir;
echo "<hr>";
if(is_dir($dir)) {
        echo "<table border='1' width='100%'>";
        echo "<tr>  <th scope='col'>Filename</th>   <th scope='col'>File Size</th>  <th scope='col'>File Type</th></tr>";
        $dirEntries = scandir($dir);
        foreach ($dirEntries as $entry) {
            echo "<tr><td>$entry</td><td>" ,
                    filesize($dir . "/" . $entry) ,
                    "</td><td>" ,
                    filetype($dir . "/" . $entry) ,
                "</td></tr>";
        }
        echo "</table>";
} else {
        echo "<p>The directory does not exist.</p>";
}

?>
```

Output of script with file and directory information functions

Obtaining File and Directory Information
(continued)


/home/staff/accounts/amolnar/cos30020/www/htdocs/l5

Filename	File Size	File Type
.	144	dir
..	4096	dir
filesdir.php	542	file
forecast.php	880	file
form_dmo.html	526	file
process_basic.php	685	file
readdir.php	192	file
sfweather.txt	224	file
slashes.php	135	file

Output of script with file and directory information functions

Copying and Moving Files

- Use the `copy()` function to copy a file with PHP
- The function returns a value of true if it is successful or false if it is not
- The syntax for the `copy()` function is:

```
copy(source, destination)
```

- For the *source* and *destination* arguments:
  - Include just the name of a file to make a copy in the current directory, or
  - Specify the entire path for each argument

Copying and Moving Files (continued)

```
if (file_exists("sfweather.txt")) {
  if(is_dir("history")) {// Windows path
       if (copy("sfweather.txt",
                "history\\sfweather01-27-2006.txt")) {
           echo "<p>File copied successfully.</p>";
       } else {
           echo "<p>Unable to copy the file!</p>";
       }
  } else {
       echo ("<p>The directory does not exist!</p>");
  }
} else {
   echo ("<p>The file does not exist!</p>");
}
```

Renaming Files and Directories

- Use the **`rename()`** function to rename a file or directory with PHP
- The `rename()` function returns a value of *true* if it is successful or *false* if it is not
- The syntax for the `rename()` function is:

```
rename(old_name, new_name)
```

Removing Files and Directories

- Use the **`unlink()`** function to delete files and the `rmdir()` function to delete directories
- Pass the name of a file to the `unlink()` function and the name of a directory to the `rmdir()` function
- Both functions return a value of *true* if successful or *false* if not
- Use the **`file_exists()`** function to determine whether a file or directory name exists before you attempt to delete it

Information for Assignment 1

On mercury

username/cos30020/www/htdocs

username/cos30020/www/data

```
.../www> chmod 0277 data
```

See Labs

For mercury File Permissions, see:
https://feenix.swin.edu.au/help/?page=Mercury%20Web%20Server

In PHP:

```
<?php
…  umask(0007);
…  mkdir($newdir, 0277);
       // create "$newdir" directory and set access
       // e.g.  $newdir="../../data/lab05/" under 'data'
```

See Labs

```
…
?>
```

SWINBURNE UNIVERSITY OF TECHNOLOGY

WORKING WITH FILES

Opening and Closing a File

- A **stream** is a channel used for accessing a resource that you can read from and write to
- The **input stream** *reads* data from a resource (such as a file)
- The **output stream** *writes* data to a resource
- Usually a three stage process:
  1. Open the file stream with the `fopen()` function
  2. Write data to or read data from the file stream
  3. Close the file stream with the `fclose()` function

Opening a File

- A **handle** is a special type of variable that PHP uses to represent a resource such as a file
- The **`fopen()`** function opens a handle to a file stream
- The syntax for the `fopen()` function is:

```
$open_file = fopen("text file", "mode");
```

file handle

- A **file pointer** is a special type of variable that refers to the currently selected line or character in a file

Question: any other types of files in addition to text file?

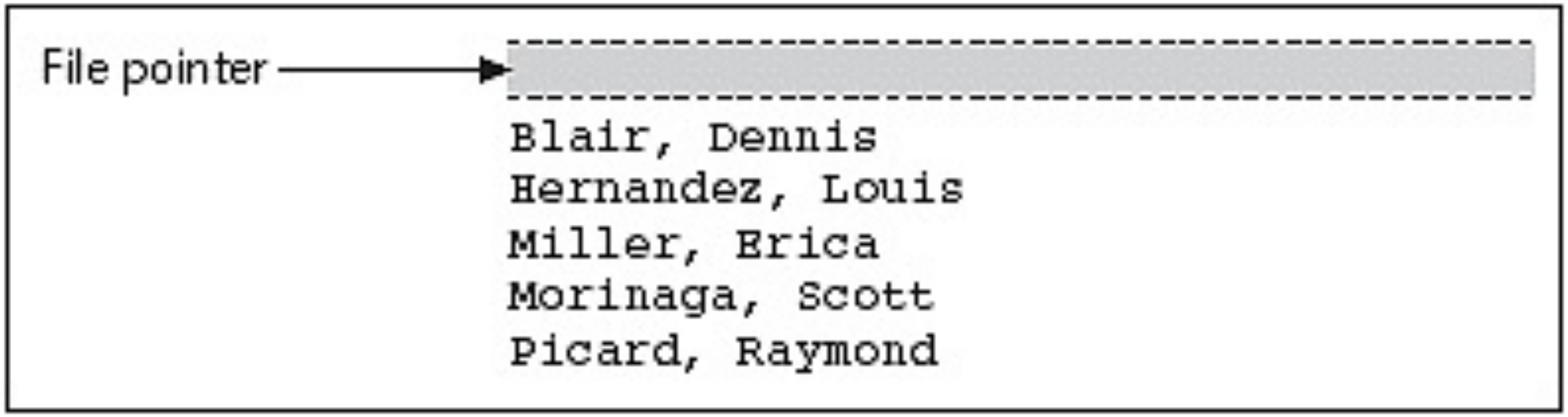
Opening a File (continued)

Mode arguments of the `fopen()` function

Argument	Description
a	Opens the specified file for writing only and places the file pointer at the end of the file; attempts to create the file if it doesn’t exist
a+	Opens the specified file for reading and writing and places the file pointer at the end of the file; attempts to create the file if it doesn't exist
r	Opens the specified file for reading only and places the pointer at the beginning of the file
r+	Opens the specified file for reading and writing and places the file pointer at the beginning of the file
w	Opens the specified file for writing only and deletes any existing content in the file; attempts to create the file if it doesn’t exit
w+	Opens the specified file for reading and writing and deletes any existing content in the file; attempts to create the file if it doesn’t exist
x	Creates and opens the specified file for writing only; returns false if the file already exists
x+	Creates and opens the specified file for reading and writing; returns false if the file already exists

Opening a File (continued)

```
$bowlersFile = fopen("bowlers.txt", "r+");
```

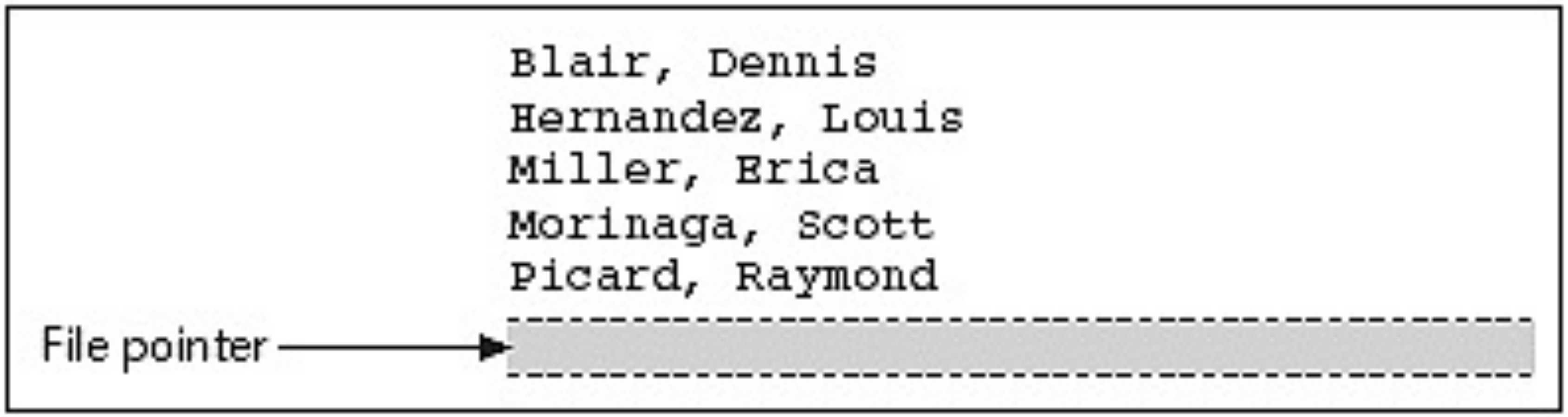


```
Blair, Dennis
Hernandez, Louis
Miller, Erica
Morinaga, Scott
Picard, Raymond
```

Location of the file pointer when the `fopen()` function uses a *mode* argument of "r+"

Opening a File (continued)

```
$bowlersFile = fopen("bowlers.txt", "a+");
```



Location of the file pointer when the `fopen()` function uses a *mode* argument of "a+"

Closing a File

- Use the `fclose` function when finished working with a file stream to save space in memory

```
$bowlersFile = fopen("bowlers.txt", "a");
$newBowler = "Doe, John\n";
:
fclose($bowlersFile);
```

Note: In these examples only a filename ("bowlers.txt") is used. In a real world example, the full relative file path and filename would be used.
For mercury help, see:
https://feenix.swin.edu.au/help/?page=Mercury%20Web%20Server

Writing Data to a File

- PHP supports two basic functions for writing data to text files:
  - `fwrite()` function *incrementally writes* data to a text file
  - `file_put_contents()` function *writes an entire file* or *appends* a text string to a file
- Escape sequences used to identify the end of a line:
  - UNIX/Linux platforms use the `\n` carriage return
  - Macintosh platforms use `\r` carriage return *(OS X is Linux based)*
  - Windows uses both the `\n` newline and the `\r` carriage return escape sequence `\n\r`

Writing Data Incrementally

- Use the **`fwrite()`** function to *incrementally write* data to a text file

  Note: **`fputs()`** is an alias for **`fwrite()`**

- The syntax for the `fwrite()` function is:

```
fwrite($handle, data[, length]);
```

- The `fwrite()` function returns the number of bytes that were written to the file
- If no data was written to the file, the function returns a value of 0

Writing Data Incrementally

```
$bowlersFile = fopen("bowlers.txt", "a");
$newBowler = "Doe, John\n";
fwrite($bowlersFile, $newBowler);
fclose($bowlersFile);
```

Writing Data Once into a File

- The **`file_put_contents()`** function *writes an entire file* or *appends* a text string to a file
- The syntax for the `file_put_contents()` function is:

```
file_put_contents (filename, string[, options])
```

Note: no file open/close needed - integrated

Writing Data Once into a File(continued)

For the 3rd argument

- The `FILE_USE_INCLUDE_PATH` constant searches for the specified filename in the path that is assigned to the `include_path` directive in your php.ini configuration file
- The `FILE_APPEND` constant appends data to any existing contents in the specified filename instead of overwriting it

`file_put_contents()` Function

```
$tournamentBowlers = "Blair, Dennis\n";
$tournamentBowlers .= "Hernandez, Louis\n";
$tournamentBowlers .= "Miller, Erica\n";
$tournamentBowlers .= "Morinaga, Scott\n";
$tournamentBowlers .= "Picard, Raymond\n";
$bowlersFile = "bowlers.txt";
file_put_contents($bowlersFile,$tournamentBowlers);
```

- If no data was written to the file, the function returns a value of 0

```
if (file_put_contents($bowlersFile,$tournamentBowlers)> 0){
   echo "<p>Data was successfully written to the
              $bowlersFile file.</p>";
} else {
   echo "<p>No data was written to the $bowlersFile
   file.</p>";
}
```

Writing Data Once into a File (continued)

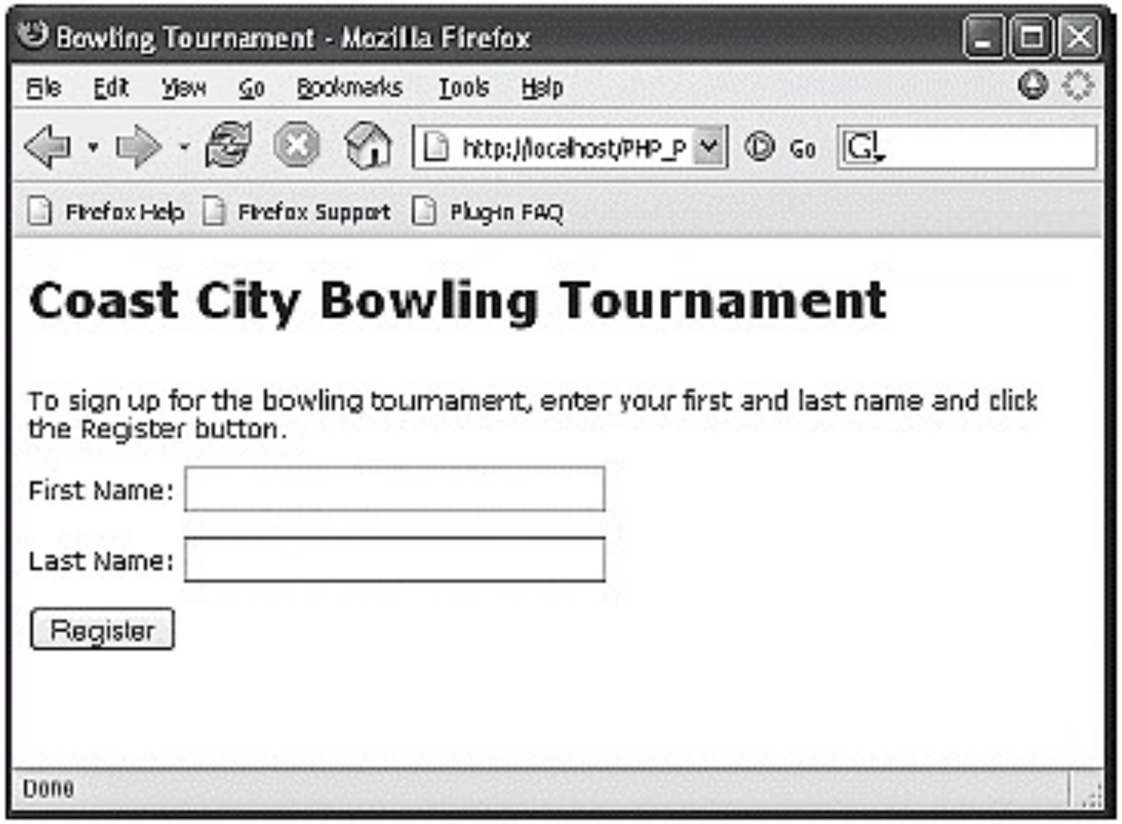
```
<h1>Coast City Bowling Tournament</h1>
<?php
if (isset($_GET["first_name"]) && isset($_GET["last_name"])) {
     $bowlerFirst = $_GET["first_name"];
     $bowlerLast = $_GET["last_name"];
     $newBowler = $bowlerLast . ", " . "$bowlerFirst" . "\n";
     $bowlersFile = "bowlers.txt";
     if (file_put_contents($bowlersFile, $newBowler, FILE_APPEND) > 0)
        echo "<p>{$_GET['first_name']} {$_GET['last_name']} has
            been registered for the bowling tournament!</p>";
     else
        echo "<p>Registration error!</p>";
} else {
     echo "<p>To sign up for the bowling tournament, enter your first
        and last name and click the Register button.</p>";
}
?>
```

Writing Data Once into a File (continued)

enctype *attribute not needed, this is the default. (Unless file upload, or text/plain)*

```
<form action="BowlingTournament.php" method="get"
enctype="application/x-www-form-urlencoded">
<p><label for="fname">First Name: </label>
   <input type="text" name="first_name" size="30" id="fname"/>
</p>
<p><label for="lname">Last Name: </label>
   <input type="text" name="last_name" size="30" id="lname"/>
</p>
<p><input type="submit" value="Register" /></p>
</form>
```

Writing Data Once into a File (continued)


Bowling Tournament - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost/PHP_P
Go
Firefox Help
Firefox Support
Plug-in FAQ

Coast City Bowling Tournament

To sign up for the bowling tournament, enter your first and last name and click the Register button.

First Name:

Last Name:

Register

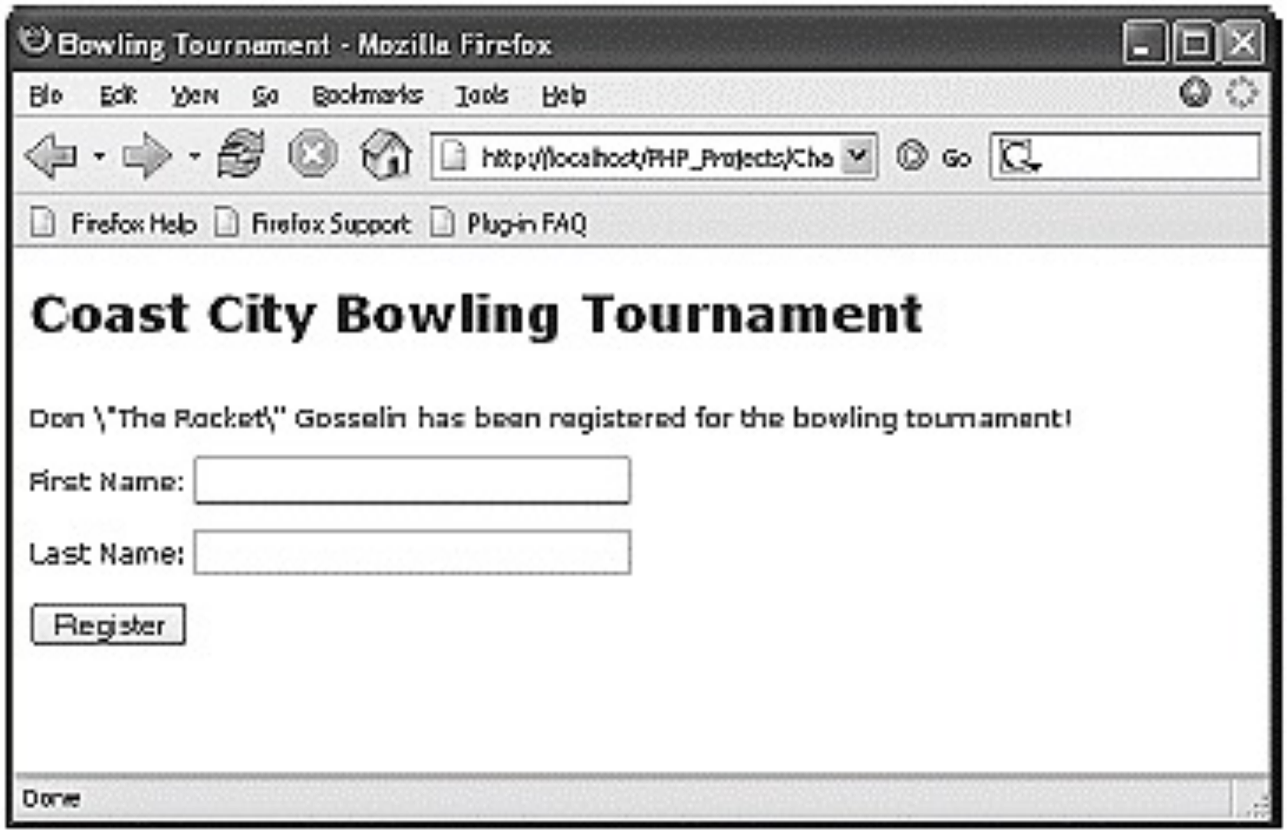
Done

Bowling registration form

addslashes() Function

```
if (isset($_GET["first_name"]) && isset($_GET["last_name"])){
      $bowlerFirst = addslashes($_GET["first_name"]);
      $bowlerLast = addslashes($_GET["last_name"]);
      $newBowler = $bowlerLast . ", " . "$bowlerFirst" . "\n";
      $bowlersFile = "bowlers.txt";
      if (file_put_contents($bowlersFile, $newBowler, FILE_APPEND) > 0)
            echo "<p>{$_GET['first_name']}{$_GET['last_name']}
                   has been registered for the bowling tournament!</p>";
      else
            echo "<p>Registration error!</p>";
} else {
      echo "<p>To sign up for the bowling tournament, enter your
             first and last name and click the Register
             button.</p>";
}
```

`addslashes()` Function (continued)


Bowling Tournament - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost/PHP_Projects/Cha
Go
Firefox Help
Firefox Support
Plug-in FAQ
Register
Done

Coast City Bowling Tournament

Don \"The Rocket\" Gosselin has been registered for the bowling tournament!

First Name:

Last Name:

Output of text with escaped characters

`stripslashes()` Function

- To prevent the display of escaped characters, use the `stripslashes()` function

```
if (file_put_contents($bowlersFile, $newBowler, FILE_APPEND) > 0)
      echo "<p>" . stripslashes($_GET["first_name"]) . " "
              . stripslashes($_GET["last_name"])
              . " has been registered for the bowling tournament!</p>";
else
      echo "<p>Registration error!</p>";
```

Reading Data to a File

- PHP supports two basic functions for reading data to text files:
  - functions *incrementally reads* data to a text file
  - functions *read an entire file* into text string variable

Reading Data Incrementally

PHP functions that iterate through a text file

Function	Description
fgetc($handle)	Returns a single character and moves the file pointer to the next character
fgetcsv($handle, length[, delimiter, string_enclosure])	Returns a line, parses the line for CSV fields, and then moves the file pointer to the next line
fgets($handle[, length])	Returns a line and moves the file pointer to the next line
fgetss($handle, length[, allowed_tags])	Returns a line, strips any HTML tags the line contains, and then moves the file pointer to the next line
stream_get_line($handle, length, delimiter)	Returns a line that ends with a specified delimiter and moves the file pointer to the next line

- The commonly used `fgets()` function uses the file pointer to iterate through a text file

Reading Data Incrementally (continued)

- You must use `fopen()` and `fclose()` with the functions listed in the table in the previous slide.
- Each time you call any of these functions, the file pointer automatically moves to the next ***line*** in the text file (except for `fgetc()`)
- Each time you call the `fgetc()` function, the file pointer moves to the next ***character*** in the file
- Often combined with the **`feof()`** function

Reading Data Incrementally (continued)

```
$handle = fopen("sfjanaverages.txt", "r");
while (! feof($handle) ) {
  $curLine = fgets ($handle);
   $curDay = explode(", ", $curLine);
     echo "<p><strong>Day " . ($i + 1)
            . "</strong><br/>";
     echo "High: {$curDay[0]}<br />";
     echo "Low: {$curDay[1]}<br />";
     echo "Mean: {$curDay[2]}</p>";
}
fclose ($handle);
```

Reading an Entire File

PHP functions that read the entire contents of a text file

Function	Description
file(filename[, use_include_path])	Reads the contents of a file into an indexed array
file_get_contents(filename[, use_include_path])	Reads the contents of a file into a string
fread($handle, length)	Reads the content of a file into a string up to a maximum number of bytes
Readfile(filename[, use_include_path])	Prints the contents of a file

- Note: no file open/close needed for `file`, `file_get_contents`, `readfile` - integrated (but not `fread` as it needs a *handle*)

`file_get_contents()` Function

- Reads the entire contents of a file into a string

```
$dailyForecast = "<p><strong>San Francisco daily
   weather forecast</strong>: Today: Partly cloudy.
   Highs from the 60s to mid 70s. West winds 5 to 15
   mph. Tonight: Increasing clouds. Lows in the mid
   40s to lower 50s. West winds 5 to 10 mph.</p>";
file_put_contents("sfweather.txt", $dailyForecast);


$sfWeather = file_get_contents("sfweather.txt");
echo $sfWeather;
```

`readfile()` Function

- If you only want to print the contents of a text file, you need not use `file_get_contents`
- Prints the contents of a text file along with the file size to a Web browser

```
readfile("sfweather.txt");
```

`file()` Function

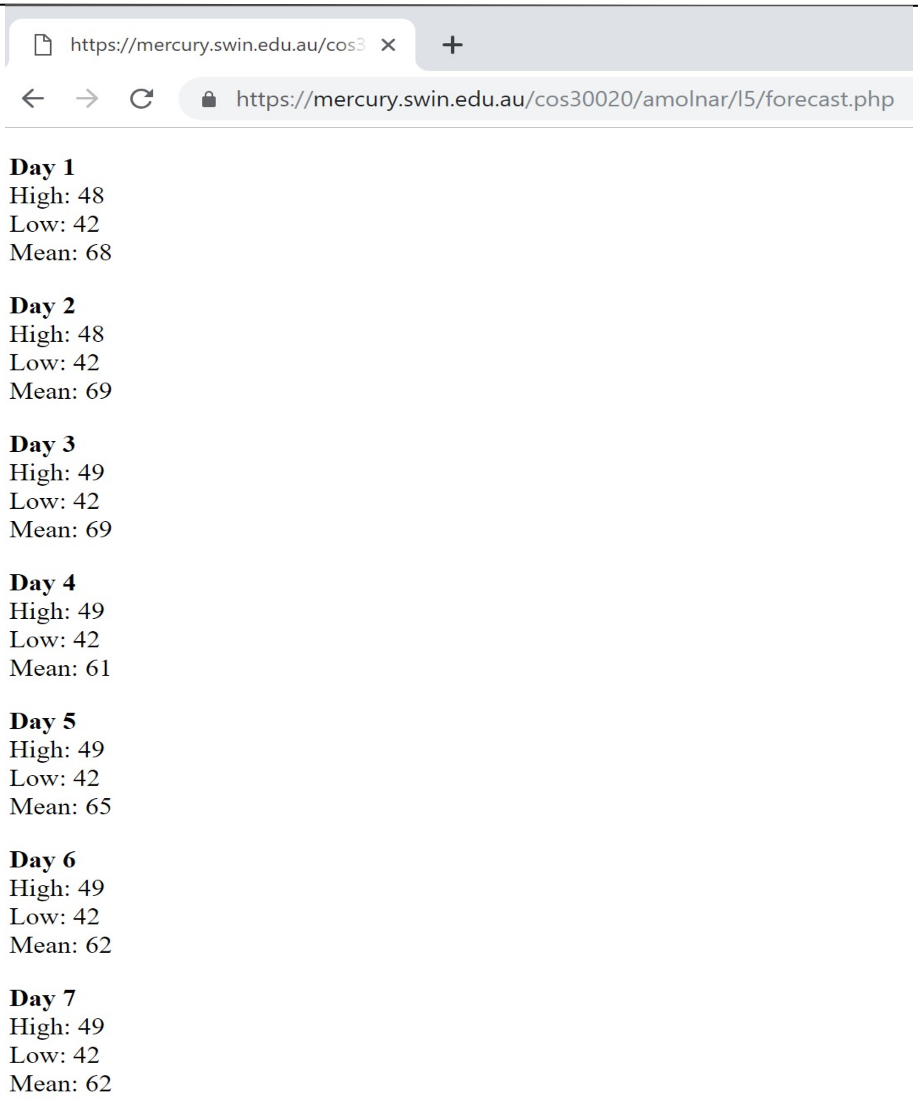
- Reads the entire contents of a file into an ***indexed array***
- Automatically recognises whether the lines in a text file end in `\n`, `\r`, or `\r\n`

```
$january = "48, 42, 68\n";
$january .= "48, 42, 69\n";
$january .= "49, 42, 69\n";
$january .= "49, 42, 61\n";
$january .= "49, 42, 65\n";
$january .= "49, 42, 62\n";
$january .= "49, 42, 62\n";
file_put_contents("sfjanaverages.txt", $january);
```

`file()` Function (continued)

```
$januaryTemps = file("sfjanaverages.txt");
for ($i=0; $i<count($januaryTemps); $i++) {
    $curDay = explode(", ", $januaryTemps[$i]);
    echo "<p><strong>Day " . ($i + 1)
           . "</strong><br/>";
    echo "High: {$curDay[0]}<br />";
    echo "Low: {$curDay[1]}<br />";
    echo "Mean: {$curDay[2]}</p>";
}
```

`file()` Function (continued)


https://mercury.swin.edu.au/cos3

https://mercury.swin.edu.au/cos30020/amolnar/l5/forecast.php

```
Day 1
High: 48
Low: 42
Mean: 68

Day 2
High: 48
Low: 42
Mean: 69

Day 3
High: 49
Low: 42
Mean: 69

Day 4
High: 49
Low: 42
Mean: 61

Day 5
High: 49
Low: 42
Mean: 65

Day 6
High: 49
Low: 42
Mean: 62

Day 7
High: 49
Low: 42
Mean: 62
```

Output of individual lines in a text file

Locking Files

- Use the **`flock()`** function, to prevent multiple users from modifying a file simultaneously
- The syntax for the `flock()` function is:

```
flock($handle, operation)
```

Operational constants of the `flock()` function

Constant	Description
LOCK_EX	Opens the file with an exclusive lock for writing
LOCK_NB	Prevents the flock() function from waiting, or “blocking”, until a file is unlocked
LOCK_SH	Opens the file a shared lock for reading
LOCK_UN	Releases a file lock

Summary

- The stream is used for accessing a resource, such as a file, that you can read from and write to
- A handle is a special type of variable that PHP uses to represent a resource such as a file
- The `fopen()` function opens a stream to a text file
- A file pointer is a special type of variable that refers to the currently selected line or character in a file
- Use the `fclose()` function to ensure that the file doesn't keep taking up space in your computer's memory

Summary (continued)

- To iterate through the entries in a directory, you open a handle to the directory with the `opendir()` function
- PHP includes various file and directory status functions, such as the `file_exists()` function, which determines whether a file or directory exists
- PHP supports two basic methods for writing data to text files: `fwrite()` and the `file_put_contents()` function
- PHP includes various functions, such as the `fgets()` function, that allow you to use the file pointer to iterate through a text file