INTRODUCTION TO PHP (CONTINUED)

Chapter 9

Note

- Examples for this chapter are at

https://swe.umbc.edu/~zzaidi1/is448/chap9-examples/

- PHP programs cannot be seen in the browser by doing a 'View Source' or 'Save as' in the browser. All PHP examples used in this chapter are zipped up as php3.zip in the above examples folder

PHP functions to Open and Close Files

Function name	Description
fopen	fopen is used to create a file handle for accessing a file. Needs as input the name of the file and the mode of access.
fclose	fclose closes a file. Needs the file handle as input

11.11 Opening and Closing Files

- The PHP function fopen is used to create a file handle for accessing a file
  - First argument is the name of the file
  - Second argument to fopen specifies the mode of access
  - The fopen function returns a file handle
- Every open file has a current pointer indicating a point in the file
- Normally input and output operations occur at the current pointer position
- The file_exists function tests if a file, given by name, exists
- The function fclose closes a file handle
- Example: See simple_file.php

```
$filevar = fopen("testdata.txt" , "r") or die ("Error - could not open file");
if(file_exists("testdata.txt")) print("file exists"); else print ("file does not exist");
fclose($filevar);
```

11.11 Modes of access

Mode	Description
"r"	Read only. The file pointer is initialized to the beginning of the file.
"r+"	Read and write an existing file. The file pointer is initialized to the beginning of the file; if a read operation precedes a write operation, the new data is written just after where the read operation left the file pointer.
"w"	Write only. Initializes the file pointer to the beginning of the file; creates the file if it does not exist.
"w+"	Read and write. Initializes the file pointer to the beginning of the file; creates the file if it does not exist. Always initializes the file pointer to the beginning of the file before the first write, destroying any existing data.
"a"	Write only. If the file exists, initializes the file pointer to the end of the file; if the file does not exist, creates it and initializes the file pointer to its beginning.
"a+"	Read and write a file, creating the file if necessary; new data is written to the end of the existing data

11.11 Reading from a File

- Most common way to read a file: get its contents into a scalar variable as a string
- Several PHP functions to help with this
- The fread function reads a given number of bytes from a file given by a file handle.
  - The entire file can be read by using the filesize function to determine the number of bytes in the file
  - This function needs the file handle as an argument
- Example: See file_reading.php

```
$filevar = fopen("testdata.txt" , "r") or die ("Error - could not open file");
if(file_exists("testdata.txt")) print("file exists"); else print ("file does not exist");
$file_content_string = fread($filevar, filesize("testdata.txt"));
fclose($filevar);
```

11.11 Reading from a File

- The file method returns an array of lines from a file named as a parameter
  - No explicit open and close are required for using this function
  - It does not use a file handle as an argument
  - It needs only the file name as an argument
- Example: See file_reading.php

```
if(file_exists("testdata.txt")) print("file exists"); else print ("file does not exist");
$file_content = file("testdata.dat");
```

11.11 Reading from a File

- The file_get_contents method returns the content of a named file as a single string
  - No explicit open and close are required for using this function
  - It does not use a file handle as an argument
  - It only needs the filename as an argument

```
if(file_exists("testdata.txt")) print("file exists"); else print ("file does not exist");
$file_content_string = file_get_contents("testdata.txt");
```

11.11 Reading from a File

- The fgets method returns a string read from file starting from current pointer position until end-of-line
- The fgetc method returns a single character
- The feof method returns TRUE if the last character read was the end of file marker, that is, the read was past the end of the file
- Example: See file_reading.php

```
$filevar = fopen("testdata.txt" , "r") or die ("Error - could not open file");
//reads contents of file until it finds a newline or eof or has read 99 chars
$line = fgets($filevar, 100);
//retrieves the character at the current pointer position
$char = fgetc($filevar);
```

PHP functions to read from a file

Function name	Description
fread	fread function reads a given number of bytes from a file. Needs a file handle as input.
file	file method returns an array of lines from a file. Needs the file name as input.
file_get_contents	file_get_contents method returns the content as a single string. Needs the file name as input.
fgets	fgets method returns a string read from file starting from current pointer position until end-of-line. Needs a file handle as input
fgetc	fgetc method returns a single character. Needs a file handle as input

11.11 Writing to a File

- If a file handle is open to for writing or appending, then the fwrite function can be used to write bytes to the file
- See write_to_file.html, writing.php

```
$filevar2 = fopen("/afs/umbc.edu/users/s/a/sampath/pub/text-files/testdata_write.txt"" , "w")
            or die ("Error - could not open file");
//writes the string in $out_data to the file and returns the number of bytes written
//this is then stored in the variable $bytes_written
$out_data = "some string to write to file";
$bytes_written = fwrite($filevar2, $out_data);
```

11.11 Writing to a File

- The `file_put_contents` function writes a given string parameter to a named file, not a file handle
- Works in PHP5, PHP7
- Identical to calling fopen(), fwrite() and fclose() successively to write data to a file.
- See write_to_file2.html, writing2.php
- To see this example at work, see See files under 'fileputcontents' folder in examples folder

```
//writes the string in $out_data to the file specified by file name names2.txt
file_put_contents("/afs/umbc.edu/users/s/a/sampath/pub/text-files/names2.txt",
                  $out_data);
```

Dealing with Concurrency

- Must be able to handle concurrent accesses to files
- Concurrency solutions
  - lock the file before a write
  - unlock the file after the write
- PHP function flock
  - used to both lock and unlock files
  - takes two parameters: file handle, and the mode of the lock

11.11 Locking Files

- The flock function will lock a named file
- The function takes a second parameter giving the mode of the lock
  - LOCK_SH - Shared lock (reader). Allow other processes to access the file
  - LOCK_EX - Exclusive lock (writer). Prevent other processes from accessing the file
  - LOCK_UN - Release a shared or exclusive lock
  - LOCK_NB - Avoids blocking other processes while locking
- Closing a file implicitly unlocks it as well

Writing to a file

- Basic steps
  - open a file (for reading/writing/appending by specifying the mode)
  - lock file
  - write/read file
  - unlock file
  - close file

Important: Location of files on GL for writing to file

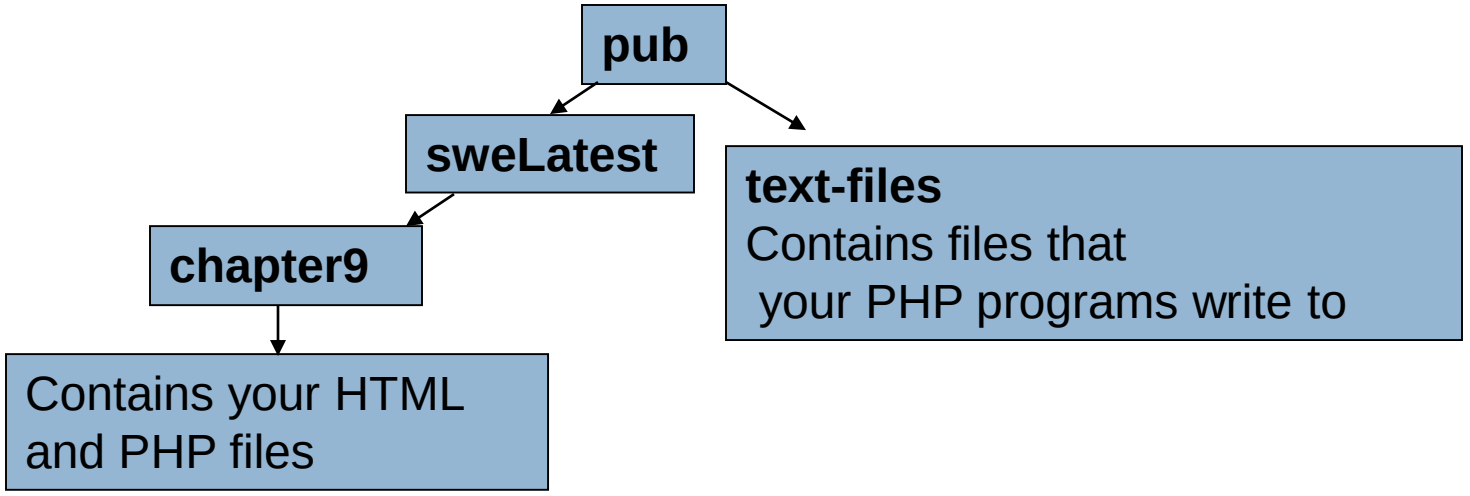
- Programs cannot write to files in the swe2019 folder
- But we want to write data to files!
- First open up Putty
- Navigate through the directories to get into the 'pub' directory
  - Type 'pwd' to see where you are currently in the directory structure
  - Use 'ls' to see contents of a directory
  - Use 'cd *dir_name*' to move into a directory by name *dir_name*
  - Use 'cd ..' to move into higher level directory
- Once you are in the 'pub' directory, create a directory called text-files under pub directory
  - Unix command to create a directory: 'mkdir text-files'
- Then, you must make text-files directory writeable by any web user. To do this, go to the directory higher than text-files and type the following at command prompt in TeraTermSSH

fs setacl text-files system:anyuser rlidwka **zzaidi1** rlidwka www.usercgi rlidwka

instead of **zzaidi1**, put your user name

- Note how all examples shown in this class are writing to a file in my text-files directory

Directory structure on GL



Lab

- Download ratings2.html
  - modify the action attribute of the form tag to point to a new PHP file
- Write a PHP file that will
  - write the user's rating to a file
    - First open the file with the appropriate access mode. Every user rating must be written to a file - think about which file mode to use.
    - Lock the file
    - Write the user-entered rating to the file using the fwrite function. You may want to write each rating on a separate line. Think about how to achieve this.
    - Unlock the file
    - Close the file
  - compute the average rating by reading all the ratings already in the file and compute the average
    - Open the file with the read mode
    - Read the contents of the file. Remember you have to do an average of the ratings, so think about which PHP function you want to use for reading and processing the file contents
    - Write the code to compute average rating. Average rating is the sum of all ratings divided by the total number of ratings in the file.
    - Print the average to the output HTML page
    - Close the file.

Solution: rating_sol2.html and rating_sol2.php