Day2

File Handling
PHP Functions
File Upload
File inclusion
Regular expressions

Files Handling

With PHP you can read from and write in files.

- The fopen() function is used to open files in PHP.
- The first parameter of this function contains the name of the file to be opened
- The second parameter specifies in which mode the file should be opened

```
<html><body>
<?php
   $file=fopen("welcome.txt","r");
?>
</body></html>
```

Files Handling

The file may be opened in one of the following modes:

r : Read only: Starts at the beginning of the file
r+ : Read/Write: Starts at the beginning of the file
w : Write only: Opens and clears the contents of file; or creates a new file if it doesn't exist
w+ : Read/Write: Opens and clears the contents of file; or creates a new file if it doesn't exist
a : Append: Opens and writes to the end of the file or creates a new file if it doesn't exist

Files Handling

File handling Functions

fopen()
fread()
fseek()
fwrite()
feof()
fclose()

PHP Functions

Syntax

```
function functionName()
{
code to be executed;
}
```

PHP function guidelines:

- Give the function a name that reflects what the function does.
- The function name can start with a letter or underscore (not a number).

PHP Functions

A function will be executed by a call to the function.

```
<html>
<body>
    <?php
        function writeName()
        {
            echo "Moataz Ahmed";
        }
        echo "My name is ";
        writeName();
    ?>
</body>
</html>
```

PHP Functions

```
<html>
  <body>
     <?php
     function add($x,$y)
     {
          $total=$x+$y;
          return $total;
     }
     echo "1 + 16 = " . add(1,16);
     ?>
  </body>
</html>
```

Output:

1 + 16 = 17

PHP Functions

PHP Built-in Functions

- Array functions
- Calendar functions
- Date functions
- Directory functions
- Error functions
- Filesystem functions
- String functions
- XML Parser functions
- Zip functions
- Mail functions
- Math functions
- MySQL functions
- SimpleXML functions

PHP Functions

Array functions

array_fill(start,number,value)

Start: A numeric value, specifies the starting index of the key
Number: A numeric value, specifies the number of entries
Value: Specifies the value to be inserted

```
<?php
$a=array_fill(2,3,“Ali");
print_r($a);
?>
```

Array ([2] => Ali [3] => Ali [4] => Ali)

PHP Functions

array_keys(Array,Value)

Array: Specifies an array.
value: You can specify a value, then only the keys with this value are returned.

```
<?php
 $a=array("a"=>“Ali","b"=>“Samy","c"=>“Ahmed");
 print_r(array_keys($a));
?>
```

```
Array ( [0] => a [1] => b [2] => c )
```

PHP Functions

array_reverse(array)

Array: Specifies an array

```
<?php
$names=array("a"=>"Ahmed","b"=>"Ali","c"=>"Ramiz");
print_r(array_reverse($names));
?>
```

Output:

```
Array ( [c] => Ramiz [b] => Ali [a] => Ahmed )
```

PHP Functions

count(array)

array: Specifies the array to count.

```
<?php
 $people = array("Rana", "Sara", "Mohamed", "Ali");
 $result = count($people);
 echo $result;
?>
```

Output: 4

PHP Functions

in_array(search,array)

search: Specifies the what to search for
array: Specifies the array to search

```
<?php
$people = array("Ali", "Ahmed", "Sami");
if (in_array("Sami",$people))
  echo "Match found";
else
 echo "Match not found";
 ?>
```

Output:
Match found

File Upload

Upload Form "upload.html"

```
<html>
  <body>
    <form action="upload_file.php" method="post"
       enctype="multipart/form-data">
     <label for="file">Filename:</label>
     <input type="file" name="myfile" id="file" />

     <input type="submit" name="submit" />
    </form>
  </body>
</html>
```

File Upload

Upload script "upload_file.php"

```
 <?php
 if ($_FILES[“myfile"]["error"] > 0)
   {
   echo "Error: " . $_FILES[“myfile"]["error"] . "<br />";
   }
 else
   {
   echo "Upload: " . $_FILES[“myfile"]["name"];
   echo "Type: " . $_FILES[“myfile"]["type"];
   echo "Size: " . ($_FILES[“myfile"]["size"];
   echo "Stored in: " . $_FILES[“myfile"]["tmp_name"];
   }
?>
```

File Upload

By using $_FILES array you can upload files from a client computer to the remote server.

The first parameter is the form's input name and the second index can be either "name", "type", "size", "tmp_name" or "error".

$_FILES[“myfile"]["name"] -name of uploaded file.
$_FILES[“myfile"]["type"] -type of uploaded file.
$_FILES[“myfile"]["size"] -size of uploaded file.
$_FILES[“myfile"]["tmp_name"] -name of temporary copy of the file stored on the server.
$_FILES[“myfile"]["error"] - the error code resulting from the file upload.

File Upload

To store the uploaded file we need to copy it to a different location:

```
<?php
   if (file_exists("upload/" . $_FILES["file"]["name"]))
   {
    echo $_FILES[“myfile"]["name"] . " already exists. ";
    } else {
      move_uploaded_file($_FILES[“myfile”]["tmp_name"],
      "upload/" . $_FILES[“myfile"]["name"]);
      echo "Stored in: " . "upload/" . $_FILES[“myfile"]
["name"];
      }?>
```

Files Inclusion

- You can include the content of a PHP file into another PHP file before the server executes it.
- There are two PHP functions which can be used to included one PHP file into another PHP file.
  1- The include() Function
  2- The require() Function

- This is a strong point of PHP which helps in creating functions, headers, footers, or elements that can be reused on multiple pages.
- This will help developers to make it easy to change the layout of complete website with minimal effort. If there is any change required then instead of changing thousand of files just change included file.

Files Inclusion

The include() Function

-The include() function takes all the text in a specified file and copies it into the file that uses the include function.

-If there is any problem in loading a file then the include() function generates a warning but the script will continue execution.

-Assume you want to create a common menu for your website.

menu.php will contain:

```
<a href="http://www.xyz.com/index.htm">Home</a> -
<a href="http://www.xyz.com/xml.html">XML</a> -
<a href="http://www.xyz.com/ajax.html">AJAX</a> -
<a href="http://www.xyz.com/perl.html">PERL</a> <br />
```

Files Inclusion

Now create as many pages as you like and include this file to create header.
For example now your test.php file can have following content.

```
<html>
  <body>
    <?php include("menu.php"); ?>
    <p>This is an example to show how to include PHP file!
</p>
  </body>
</html>
```

Output:

Home - XML - AJAX – PERL
This is an example to show how to include PHP file!

Files Inclusion

The require() Function

The require() function takes all the text in a specified file and copies it into the file that uses the include function. If there is any problem in loading a file then the require() function generates a fatal error and halt the execution of the script.

```
<html>
  <body>
    <?php   require("xxmenu.php");   ?>
    <p>This will show how to include wrong PHP file!</p>
  </body>
</html>
```

Output:

Fatal error: require() [function.require]:

Include Or Require every-time ?

Regular Expression

- The regular expression, as a pattern, can match all kinds of text strings helping your application validate,compare, compute, decide etc.

- You can match phone numbers, email addresses, url's, credit card numbers, social security numbers, zip codes, states, cities ,....

- To do a quick summary so far, a regular expression is a sequence of literal characters, wildcards, modifiers and anchors.

Regular Expression

Literal Characters

-An inclusion range [m-n] matches one of any character included in the range from m to n.
Example '[a-z]' will match any alpha character that falls within the a to z range.

-An exclusion range [^m-n] matches one of any character not included in the range from m to n.
Example '[^0-9]' will match any non-digit character.

- A period "." matches any character. It is also known as the wildcard.
Example 'a.c' will match 'aec', 'acc', 'a@a' and so on.

Regular Expression

Wildcards

The expression [:alnum:] will match all alpha-numeric characters. It is a shortcut to [A-Za-z0-9].
The expression [:alpha:] will match all alpha characters. It is a shortcut to [A-Za-z].
The expression [:blank:] will match a space or tab.
The expression [:digit:] will match a numeric digit. It is a shortcut to [0-9].
The expression [:lower:] will match all lowercase letters. It is a shortcut to [a-z].
The expression [:upper:] will match all uppercase letters. It is a shortcut to [A-Z].
The expression [:punct:] will match all printable characters, excluding spaces and alphanumerics.
The expression [:space:] will match a whitespace character.

Regular Expression

Modifiers

A modifier alters the meaning of the immediately preceding pattern character.

-**An asterisk (‘*’)** matches 0 or more of the preceding term.
Example ‘a*’ will match ”, ‘a’, ‘aa’, ‘aaaaa’ and so on.

-**A question mark (‘?’)** matches 0 or 1 of the preceding term.
Example ‘a?’ will match ” and ‘a’ only.

-**A plus sign (‘+’)** matches 1 or more of the preceding term.
Example ‘a+’ will match ‘a’, ‘aaaa’ and so on.

-**{m,n}** matches between m and n occurences of the preceding term.
Example ‘a{1,3}’ will match ‘a’, ‘aa’ and ‘aaa’ only.

-**{n}** matches exactly n occurences of the preceding term.
Example ‘a{2}’ will match ‘aa’ only.

Regular Expression

Anchors

Anchors establish the context for the pattern such as “the beginning of a word” or “end of word”.

-The pike ‘^’ marks the beginning of a line.
Example ‘^http’ will match any new line that starts with ‘http’.

-The dollar sign ‘$’ marks the end of a line.
Example ‘after$’ will match any line that ends with ‘after’.

Regular Expression

Regex quick reference

[abc] A single character: a, b or c
[^abc] Any single character but a, b, or c
[a-z] Any single character in the range a-z
[a-zA-Z] Any single character in the range a-z or A-Z
^ Start of line
$ End of line
\A Start of string
\z End of string
. Any single character
(a|b) a or b
a? Zero or one of a
a* Zero or more of a
a+ One or more of a
a{3} Exactly 3 of a
a{3,} 3 or more of a
a{3,6} Between 3 and 6 of a