Web Programming

YJ – Aug 2015

Course Outline

There're 16 lectures (2.5 hrs each) in this course. We'll mainly focus on web programming using LAMP model.

During the 24 hours tutorials in each week, our tutors will help you go through all the contents in the lecture notes and help you with the assignments

All scheduled lectures are learner-paced which means they could be altered depends on the situation. We help you gain rather than cram!

Course Outline

- L1 ~ Web programming intro
- L2 ~ HTML5
- L3 ~ CSS3
- L4 ~ JavaScripts
- L5 ~ PHP 1 Basic
- L6 ~ PHP 2 OO && Functions
- L7 ~ PHP 3 OO && Sessions
- L8 ~ PHP 4 Advanced features
- L9 ~ Mysql
- L10 ~ Mysql 2
- L11 ~ XML/JSON
- L12 ~ Ajax
- L13 ~ Framework: Bootstrap && Laravel
- L14 ~ CMS: Wordpress, Joomla, etc.
- L15 ~ Final Projects
- L16 ~ Final Projects

File Access

Open a file

```
$filename = "/home/user/guest/tmp.txt";
$file = fopen( $filename, "r" );
```

Mode	Purpose
r	Opens the file for reading only.Places the file pointer at the beginning of the file.
r+	Opens the file for reading and writing.Places the file pointer at the beginning of the file.
w	Opens the file for writing only.Places the file pointer at the beginning of the file. and truncates the file to zero length. If files does not exist then it attemts to create a file.
w+	Opens the file for reading and writing only.Places the file pointer at the beginning of the file. and truncates the file to zero length. If files does not exist then it attemts to create a file.
a	Opens the file for writing only.Places the file pointer at the end of the file. If files does not exist then it attemts to create a file.
a+	Opens the file for reading and writing only.Places the file pointer at the end of the file. If files does not exist then it attemts to create a file.

File Access

Read a file

```
$filename = "/home/user/guest/tmp.txt";
$file = fopen( $filename, "r" );
$filesize = filesize( $filename );
$filetext = fread( $file, $filesize );
```

Write a file

```
fwrite( $file, "This is  a simple test\n" );
```

Close a file

```
fclose( $file );
```

Upload Files

The process of uploading a file follows these steps

- The user opens the page containing a HTML form featuring a text files, a browse button and a submit button.
- The user clicks the browse button and selects a file to upload from the local PC.
- The full path to the selected file appears in the text filed then the user clicks the submit button.
- The selected file is sent to the temporary directory on the server.
- The PHP script that was specified as the form handler in the form's action attribute checks that the file has arrived and then copies the file into an intended directory.
- The PHP script confirms the success to the user.

There is one global PHP variable called **$_FILES**. This variable is an associate double dimension array and keeps all the information related to uploaded file.

- **$_FILES['file']['tmp_name']-** the uploaded file in the temporary directory on the web server.
- **$_FILES['file']['name'] -** the actual name of the uploaded file.
- **$_FILES['file']['size'] -** the size in bytes of the uploaded file.
- **$_FILES['file']['type'] -** the MIME type of the uploaded file.
- **$_FILES['file']['error'] -** the error code associated with this file upload.

REGULAR EXPRESSIONS

- Regular expressions are nothing more than a sequence or pattern of characters itself. They provide the foundation for pattern-matching functionality.
- Using regular expression you can search a particular string inside a another string, you can replace one string by another string and you can split a string into many chunks.
- Very useful for text processing applications – many web applications fit this pattern.
- For example, an email address follows a regular pattern.
- To check if an email address entered by a user is valid, we can check it against a regular expression that describes the pattern followed by email addresses.

REGULAR EXPRESSIONS

Expression	Description
[0-9]	It matches any decimal digit from 0 through 9.
[a-z]	It matches any character from lowercase a through lowercase z.
[A-Z]	It matches any character from uppercase A through uppercase Z.
[a-Z]	It matches any character from lowercase a through uppercase Z.
[^a-zA-Z]	It matches any string not containing any of the characters ranging from a through z and A through Z.

REGULAR EXPRESSIONS

Expression	Description
p+	It matches any string containing at least one p.
p*	It matches any string containing zero or more p's.
p?	It matches any string containing zero or more p's. This is just an alternative way to use p*.
p{**N**}	It matches any string containing a sequence of **N** p's
p{2,3}	It matches any string containing a sequence of two or three p's.
p{2, }	It matches any string containing a sequence of at least two p's.
p$	It matches any string with p at the end of it.
^p	It matches any string with p at the beginning of it.

REGULAR EXPRESSIONS

Expression	Description
p.p	It matches any string containing p, followed by any character, in turn followed by another p.
^.{2}$	It matches any string containing exactly two characters.
(.*)	It matches any string enclosed within and .
p(hp)*	It matches any string containing a p followed by zero or more instances of the sequence hp.

REGULAR EXPRESSIONS

Expression	Description
[[:alpha:]]	It matches any string containing alphabetic characters aA through zZ.
[[:digit:]]	It matches any string containing numerical digits 0 through 9.
[[:alnum:]]	It matches any string containing alphanumeric characters aA through zZ and 0 through 9.
[[:space:]]	It matches any string containing a space.
[[:upper:]]	It matches an uppercase character
[[:punct:]]	It matches a punctuation character

REGULAR EXPRESSIONS

Character	Description
.	a single character
\s	a whitespace character (space, tab, newline)
\S	non-whitespace character
\d	a digit (0-9)
\D	a non-digit
\w	a word character (a-z, A-Z, 0-9, _)
\W	a non-word character
[aeiou]	matches a single character in the given set
[^aeiou]	matches a single character outside the given set
(foo|bar|baz)	matches any of the alternatives specified

REGULAR EXPRESSIONS

You can give choices using the pipe character | (read it as OR)

For example:
(com) | (edu) | (org) | (net)
matches com or edu or org or net

If you want to match a character that has some special meaning
(.* + [ ]$^())

you need to say "I mean a literal one of these, not the special meaning".
Put a backslash in front of it: \
If you want to match a backslash, you need two backslashes: \\

REGULAR EXPRESSIONS

Function	Description
preg_match()	The preg_match() function searches string for pattern, returning true if pattern exists, and false otherwise.
preg_match_all()	The preg_match_all() function matches all occurrences of pattern in string.
preg_replace()	The preg_replace() function operates just like ereg_replace(), except that regular expressions can be used in the pattern and replacement input parameters.
preg_split()	The preg_split() function operates exactly like split(), except that regular expressions are accepted as input parameters for pattern.
preg_grep()	The preg_grep() function searches all elements of input_array, returning all elements matching the regexp pattern.

Error Handling

die()

```php
<?php
if(!file_exists("/tmp/textNoExist.txt"))
{
   die("File not found");
}
else
{
   $file=fopen("/tmp/test.txt","r");
   print "Opend file sucessfully";
}
// Test of the code here.
?>
```

try…catch

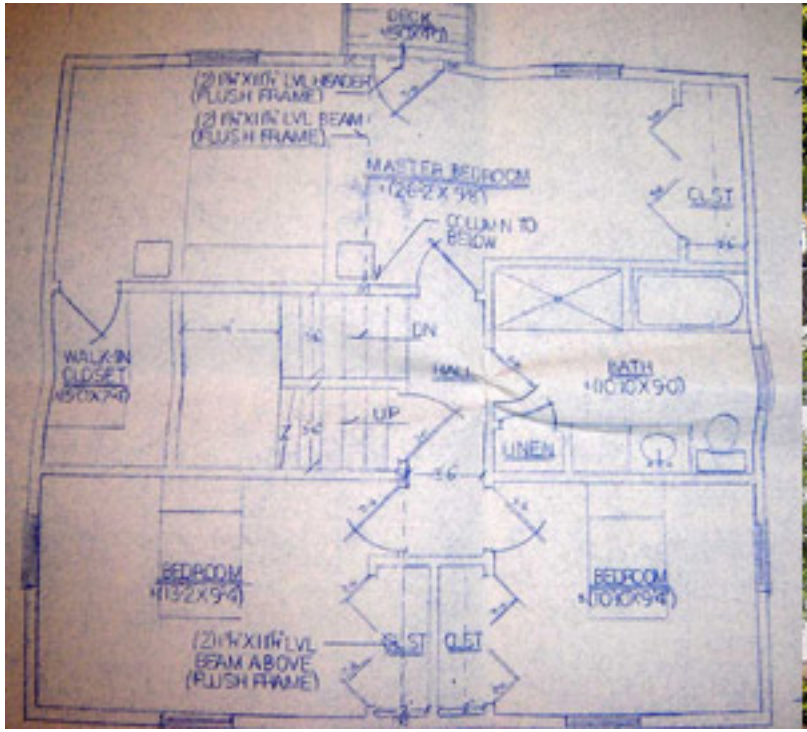
```php
<?php
try {
   $error = 'Always throw this error';
   throw new Exception($error);

   // Code following an exception is not executed.
   echo 'Never executed';

} catch (Exception $e) {
   echo 'Caught exception: ', $e->getMessage(), "\n";
}

// Continue execution
echo 'Hello World';
?>
```

Function

```php
<?php
/* Defining a PHP Function */
function writeMessage()
{
   echo "You are really a nice person, Have a nice time!";
}
/* Calling a PHP Function */
writeMessage();
?>
```

Object-Oriented

- Object-oriented programming is a style of coding that allows developers to group similar tasks into classes. This helps keep code following the tenet "don't repeat yourself" (DRY) and easy-to-maintain.



Class

Class is a programmer-defined datatype, which includes local functions as well as local data. You can think of a class as a template for making many instances of the same kind (or class) of object.

```php
<?php

class MyClass
{
    // Class properties and methods go here
}

$obj = new MyClass;

var_dump($obj);

?>
```

Properties

Properties are the variables defined inside a class. This data will be invisible to the outside of the class and can be accessed via member functions. These variables are called attribute of the object once an object is created.

```php
<?php

class MyClass
{
    public $prop1 = "I'm a class property!";
}

$obj = new MyClass;

echo $obj->prop1; // Output the property

?>
```

Methods

Methods are the function defined inside a class and are used to access object data.

```php
<?php
class MyClass
{
   public $prop1 = "I'm a class property!";

   public function setProperty( $newval )
   {
      $this->prop1 = $newval;
   }

   public function getProperty()
   {
      return $this->prop1 . "<br />";
   }
}
$obj = new MyClass;
echo $obj->getProperty(); // Get the property value

// Set a new one
$obj->setProperty( "I'm a new property value!" );
// Read it out again to show the change
echo $obj->getProperty();
?>
```

Constructor & Destructors

When an object is instantiated, it's often desirable to set a few things right off the bat. To handle this, PHP provides the magic method __construct(), which is called automatically whenever a new object is created.

To call a function when the object is destroyed, the __destruct() magic method is available. This is useful for class cleanup (closing a database connection, for instance).

```php
<?php
class MyClass
{
   public $prop1 = "I'm a class property!";

   public function __construct()
   {
      echo 'The class "', __CLASS__, '" was initiated!<br />';
   }

   public function setProperty($newval)
   {
      $this->prop1 = $newval;
   }

   public function getProperty()
   {
      return $this->prop1 . "<br />";
   }
}
// Create a new object
$obj = new MyClass;
// Get the value of $prop1
echo $obj->getProperty();
// Output a message at the end of the file
echo "End of file.<br />";
?>
```

Inheritance

When a class is defined by inheriting existing function of a parent class then it is called inheritance. Here child class will inherit all or few member functions and variables of a parent class.

```
class MyOtherClass extends MyClass
{
   public function newMethod()
   {
      echo "From a new method in " . __CLASS__ . ".<br />";
   }
}
```

Overriding

Function definitions in child classes override definitions with the same name in parent classes. In a child class, we can modify the definition of a function inherited from parent class.

```
class MyOtherClass extends MyClass
{
    public function __construct()
    {
        parent::__construct(); // Call the parent class's constructor
        echo "A new constructor in " . __CLASS__ . ".<br />";
    }

    public function newMethod()
    {
        echo "From a new method in " . __CLASS__ . ".<br />";
    }
}
```