ITMD 462/562
Web Site Application Development

Lecture 4

Fall 2015 – September 16, 2015

Tonight's Agenda

- Error Reporting
- Using external PHP Files
- Classes and OOP
- Namespacing

Error Reporting

Displaying Errors

- You may notice that your pages do or do not output and display errors based on the configuration
- Often set in php.ini
- Settings can be overridden in your script

```
ini_set('display_errors', 1);

error_reporting(E_ALL);
```

- http://php.net/manual/en/function.error-reporting.php

Using external files

Why put everything in a single file?

External File

- We need to extract some of our code to external files
- This allows us to reuse code and better maintainability
- Common Code or Class Files are typical

```
include
require
include_once
require_once
```

- http://php.net/manual/en/function.include.php

Classes, Objects, and OOP

Classes and Objects

- Object Oriented Programming (OOP) is a pattern that abstracts and groups similar functionality into classes.
- Classes and Objects are custom data types in your application that typically represent something.
- Classes are the blueprint or template for the object
- Objects are the instantiated concrete item produced from the class
- Greatly increases maintainability and reuse of components (DRY)
- http://php.net/manual/en/language.oop5.php

Class Definition

- Class definitions begin with the class keyword, then the name of the class, followed by curly braces enclosing the contents of the class
- Can not be a reserved word and must follow standard naming. Start with a letter or underscore and then can contain letters, numbers, or underscores.
- Classes contain
  - Constants
  - Variables – called properties in classes
  - Functions – called methods in classes

Class Definition

- By convention Class names typically start with a capital letter and then camelcase

```
ExampleClass
```

- Classes have an internal `$this` variable that references the current object.

```
class SampleClass {
  // properties and methods go here
}
```

- Instantiated with the `new` keyword

```
$obj = new SampleClass();
```

http://php.net/manual/en/language.oop5.basic.php

Class Properties

- Class member variables are called properties
- Must be defined with one of the keywords `public`, `private`, `protected`, `static`
- May be initialized when defined with a value (no variable)
- Non-static properties are accessed with the arrow/object operator

  `$this->property` or `$obj->property`
- Static properties are accessed with the double colon operator

  `self::$property` or `MyClass::$property`
- http://php.net/manual/en/language.oop5.properties.php

Class Constants

- Defined with the `const` keyword. Not the define function.
- Do not use the `$` symbol

```
const MY_CONSTANT = 'something';
```

- Accessed with the double colon operator

```
self::MY_CONSTANT    or    MyClass::MY_CONSTANT
```

Visibility Keywords

- Three visibility keywords which must be used on properties and may be used on methods.
- Methods without a visibility keyword will be considered public

```
public
```

```
private
```

```
protected
```

- http://php.net/manual/en/language.oop5.visibility.php

Class Methods

- Functions defined within a class
- Can use visibility keywords, defaults to public
- `$this` variable is available to reference the current object instance.

```
class MyClass {
    public function myMethod() {
        echo ‘my method’;
    }
}

$obj = new MyClass();

$obj->myMethod(); // prints my method
```

Constructors, Destructors, and Magic Methods

- Constructor method for a class is defined with the 2 underscore construct name

```
function __construct(){
}
```

- Is called when the object is instantiated
- http://php.net/manual/en/language.oop5.decon.php
- Other magic methods exist like __toString
- http://php.net/manual/en/language.oop5.magic.php

Inheritance

- The is a way to extend another class
- Subclass inherits all the public and protected methods from parent
- It allows you to override methods in the parent class
- Uses the extends keyword

```
class Bar extends Foo {
}
```

- http://php.net/manual/en/language.oop5.inheritance.php

Static Keyword

- If you use the static keyword when declaring a property or method it is available as a class property or method without instantiating the object
- Access with the double colon (Scope Resolution Operator)

```
print Foo::$my_static
```

- http://php.net/manual/en/language.oop5.static.php

Abstract Class, Interface, Traits

- Classes defined as abstract can not be instantiated
- http://php.net/manual/en/language.oop5.abstract.php

- Interfaces are code which specifies what methods a class must implement
- http://php.net/manual/en/language.oop5.interfaces.php

- Traits, >= 5.4, a way for code reuse in single inheritance languages, like a class but can't be instantiated
- http://php.net/manual/en/language.oop5.traits.php

Final Keyword

- Prevents child subclasses from overriding a method.
- If the class is declared final it can not be sub classed.
- http://php.net/manual/en/language.oop5.final.php

Much More

- Please read through the API docs classes and objects section
- http://php.net/manual/en/language.oop5.php

Namespace

Too many classes and methods in the global namespace

Namespaces

- Namespaces are a way to organize code so everything is not defined in the global namespace.
- Similar to a Java package
- Solution for problems with naming collisions
- Uses the `namespace` keyword to define the namespace for the following code
- Uses the backslash character `\` as a separator
- When using namespaced code you can use the `use` keyword to alias the namespace
- http://php.net/manual/en/language.namespaces.php

Namespace Tutorials

- Demo
- Check out these resources for additional info
- https://knpuniversity.com/screencast/php-namespaces-in-120-seconds
- http://daylerees.com/php-namespaces-explained/
- http://code.tutsplus.com/tutorials/namespacing-in-php--net-27203
- http://www.sitepoint.com/php-53-namespaces-basics/

Assignments

Reading

- Posted in blackboard in the assignments section.
- Read object oriented PHP for beginners tutorial
- Read through php.net documentation on classes and objects that we did not cover.
- Read some of the namespace documentation links listed on the previous slide. You need to understand the namespace concept. At a minimum watch the PHP namespaces in 120 seconds video.
- https://knpuniversity.com/screencast/php-namespaces-in-120-seconds

Assignments

- Assignment 1 is Due this weekend
- Due **Sunday, September 20, 2015 at 11:59pm Chicago Time**
- We will discuss in class on September 23 so no submissions will be accepted after 6:00pm on September 23.