PHP OBJECTS

By Troy Connor

What is an object?

- **Class:** This is a programmer-defined datatype, which includes local functions as well as local data

- **Object:** An individual instance of the data structure defined by a class. You define a class once and then make many objects that belong to it. Objects are also known as instance

```
<?php
        class SimpleClass{
            public $var = 'a default value';

            public function displayVar() {
             echo $this->var;
                }
        }
 $obj = new SimpleClass();
?>
```

This is an object

What does an object have?

- **Properties:** They are defined by using one of the keywords **public, protected, or private,** followed by a normal variable declaration.

- **Methods/Member function:** These are the functions defined inside a class and are used to access object data.

```
<?php
        class SimpleClass{
            // property declaration
            public $var = 'a default value';

            // method declaration
            public function displayVar() {
                echo $this->var;
                    }
        }
 $obj = new SimpleClass();
?>
```

Public, Private, and Protected

- Class members declared public can be accessed everywhere
- Members declared protected can be accessed only within the class itself and by inherited and parent classes
- Members declared as private may only be accessed by the class that defines the member
- Class methods may be defined as public, private, or protected. Methods declared without any explicit visibility keyword are defined as public

```php
<?php
class MyClass
{
    public $public = 'Public';
    protected $protected = 'Protected';
    private $private = 'Private';
    function printHello(){
        echo $this->public;
        echo $this->protected;
        echo $this->private;
    }
}
class MyClass2 extends MyClass{
                protected $protected = 'Protected2';
    function printHello()    {
        echo $this->public;
        echo $this->protected;
        echo $this->private;}
}
$obj2 = new MyClass2();
echo $obj2->public; // Works
echo $obj2->protected; // Fatal Error
echo $obj2->private; // Undefined
$obj2->printHello(); // Shows Public, Protected2, Undefined
?>
```

Object Inheritance

- Object inheritance is done by the keyword ***extends***
- The subclass inherits all of the public and protected methods from the parent class. Unless a class overrides those methods, they will retain their original functionality
- Multiple Inheritance is not supported in PHP 5
- Multilevel inheritance is allowed

```
class grandParent{
//Body of your class
}

class parent extends grandParent{
//Body of your class
}

class child extends parent{
//Body of your class
}
```

Object Interface

- Object interfaces allow you to create code which specifies which methods a class must implement, without having to define how these methods are handled.
- Interfaces are defined using the interface keyword, in the same way as a standard class, but without any of the methods having their contents defined.
- All methods declared in an interface must be public, this is the nature of an interface.
- To implement an interface, the ***implements*** operator is used

Interface example

```php
<?php
interface iTemplate
{
    public function setVariable($name, $var);
    public function getHtml($template);
}

class Template implements iTemplate
{
    private $vars = array();

    public function setVariable($name, $var)
    {
        $this->vars[$name] = $var;
    }

    public function getHtml($template)
    {
        foreach($this->vars as $name => $value) {
            $template = str_replace('{' . $name . '}', $value, $template);
        }
        return $template;
    }
}
?>
```

Magic Methods

- __construct()

  Classes which have a constructor method call this method on each newly-created object

- __destruct()

  The destructor method will be called as soon as there are no other references to a particular object

- **__toString()**

  Allows a class to decide how it will react when it is treated like a string.

- PHP reserves all function names starting with __ as magical

Scope Resolution Operator

- The Scope Resolution Operator is a token that allows access to static, constant, and overridden properties or methods of a class

```
<?php
class MyClass{
    protected function myFunc() {
        echo "MyClass::myFunc()\n";    }
}
class OtherClass extends MyClass{
    public function myFunc(){     // Override parent's definition
          parent::myFunc();        // But still call the parent function
        echo "OtherClass::myFunc()\n";
     }
}
$class = new OtherClass();
$class->myFunc();

?>
```

Static Keyword

- Declaring class properties or methods as static makes them accessible without needing an instantiation of the class. A property declared as static can not be accessed with an instantiated class object (though a static method can)

```
<?php
class Foo {
    public static function aStaticMethod() {
        // ...
    }
}

Foo::aStaticMethod();
$classname = 'Foo';
$classname::aStaticMethod(); // As of PHP 5.3.0
?>
```

Final Keyword

- The final keyword, which prevents child classes from overriding a method by prefixing the definition with *final*. If the class itself is being defined final then it cannot be extended

```
<?php
class BaseClass {
   public function test() {
       echo "BaseClass::test() called\n”;
   }
   final public function moreTesting() {
       echo "BaseClass::moreTesting() called\n";
   }
}
class ChildClass extends BaseClass {
   public function moreTesting() {
       echo "ChildClass::moreTesting() called\n";
   }
}
// Results in Fatal error: Cannot override final method BaseClass::moreTesting()
```

- ?>

Common PHP objects usage

- Model View Controller (MVC) - software architecture, or design pattern, that is used in software engineering, whose fundamental principle is based on the idea that the logic of an application should be separated from its presentation

- PDO (PHP DATA OBJECTS) – consistent interface used to access databases in PHP

- PHPMailer - provides a package of functions to send email. The two primary features are sending HTML Email and e-mails with attachments

Benefits of Objects in PHP

- Ease of implementation
- Better organization
- Easier maintenance
- Code reuse

References

- http://www.php.net
- https://code.google.com/a/apache-extras.org/p/phpmailer/wiki/UsefulTutorial
- http://code.tutsplus.com/tutorials/object-oriented-php-for-beginners--net-12762

Questions?