Object Oriented PHP

By: Ashuqullah Alizai
Herat University
Computer Science Faculty
0795642400
Alizai.csf@hotmail.com

Object oriented vs. Procedural programming

- **Procedural Programming**
  - A procedural program is traditionally a list of instructions that are followed in order, using control flow statements, functions, and so on.
  - In procedural programming, you have data (stored in variables) that you pass to functions, which perform operations on the data and may modify it or create new data.
  - Procedural programming consists of codes with or without routines.

Object oriented programming

- OOP or Object Oriented Programming
  - OOP means using the concept of Modular programming and working with objects
  - The idea of modular (or component) programming is based on the methodology of blocks of code that can be created individually to be assembled with other modules to produce a working application
  - OOP enlightens any language for better coding, for best performance and for writing very big projects without worrying a lot about managing them.
  - OOP removes the hassles and difficulties of writing and managing big applications.

Object oriented programming

- OOP is not only creating OBJECT and CLASSES
  - It change the design of web application to three tier modules that give the ability to reuse tiers with multiple applications.
  - Each tier can be independently changed and built (compiled) without affecting the other tiers.

Interface Tier

- The interface tier contains all graphics and program code related to displaying information to the user.

  - GUI
  - Information formatting
  - Information display
  - Information input from user
  - Information verification
  - Information output to BR
  - Web Page, App Form

Business rules tier

- It processes any information submitted from the interface tier and can then submit information to the data tier to be stored.

  - Data received from IT
  - Data retrieved from DT
  - Data verification
  - Data formatting
  - Information output to IT
  - Information output to DT
  - Process data into Information
  - Service or Web Service

Data tier

- *The data tier is the primary storage location for the application, which may include the use of a database*

  - Information Received from BR
  - Data received from DBMS
  - Data formatting
  - Data output to BR
  - Information verification
  - Information output to DBMS

Advantage of Object Oriented Programming

- **Re-Usability of your code**: For example, if you have created calculator class at one place then you can use the same calculator class in your application.
- **Easy to Maintain** : Application develop using oop technique are easier to maintain than normal programming.
- **Good Level of Abstraction**: Abstraction means making something hidden. By using oop technique you are abstracting your business logic from interface and data tier.
- **Modularity**: If you are creating separate class for your every problem then you are making it modular. So if someone need to change in the business logic part then he will always go to your business logic code part.

A Little History of OOP in PHP

- When PHP was developed, it did not implement OO features in itself.
- Zeev, Rasmus, and Andy rewrote the core and released PHP3, very basic OO features were introduced.
- When PHP4 was released, OO features got matured with huge performance improvement.
- PHP team rewrote the core engine again to introduce completely new object models and released PHP5.
- Now there are three versions of PHP being developed.
- PHP4, PHP5, PHP7 which php4 is almost deprecated
- PHP7 doesn't mean it is the latest PHP version.
- PHP4 and PHP5 are being released actively (though there will be no more releases of PHP4 after December 2007).
- Between these three, PHP5 and PHP7 implements almost complete OO features while PHP4 doesn't.

PHP7 performance

- With the release of the PHP 7 environment, great improvements have taken place. The new methode of encryption has been introduced. the newest PHP encryption tool “password hash” is more secured instead of MD5 that has proven to be vulnerable to hacking.
- *“PHP 7 is based on the PHPNG project (PHP Next-Gen), that was led by Zend to speed up PHP applications. The performance gains realized from PHP 7 are huge! They vary between 25% and 70% on real-world apps, and all of that just from upgrading PHP, without having to change a single line of code!”*
- PHP 7 also replaces fatal errors, which previously would crash a program, with exceptions that can be handled within the program itself.

What is Object?

- Any thing in the world is an object.
  - Your laptop, pc, car every thing is an object.
- Objects have two things(an example of a car)
  - **Properties** : Your car has property (color, brand name)
  - **Behaviors:** it can go forward and backward

- If you are able to find properties and behaviors of real object. Then it will be very easy for you to work with Object Oriented Programming.

- If you are functional programming background than properties mean variables and Behaviors mean functions

Object in Programming

- *Objects are blocks of code that have already been compiled for use within an application.*
- Every programming object has some properties and behaviors.
  - For example if you have object for calculator then it has property size, color etc... and has some behaviors and method like addition, multiplication etc.
- Is it similar to an array, as arrays can store data identified by properties (well, they are called keys)?
  - Objects are much more than arrays because they contain some methods inside them. They can either hide them or expose them, which are not possible in arrays.
- The object is somewhat comparable with a data structure, and can incorporate a lot of other objects in itself.

Some Basic OO Terms

- A class is similar to a blueprint of a house.
- The blueprint contains
  - a description (characteristics) of all the elements needed to construct the house. However, the blueprint is not the actual house itself. It describes what is possible if we hire a crew and construct the house.
  - The blueprint is not considered to exist (as a house would exist).
  - However, it describes the items needed to build the house (nails, drywall, and wood) and the process to build the house.
- A class is a template for an object.
  - A class contains the code which defines how an object will behave and interact with each other. It describes the characteristics (properties) of the module of code and the *actions* (methods or functions) that can occur in that code. However, it does not physically exist (within memory) until an instance of the class (called an object) is created. Once an instance is created, the characteristics and methods can be accessed.

Some Basic OO Terms

- **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.
- **Property:** A property is a container inside the class which can retain some information.
  - In essence, a property is a variable which is declared inside the class itself, but not inside any function in that class.
  - Unlike other languages, PHP doesn't check the type of property variable.
  - A property could be accessible only in class itself, by its subclass, or by everyone.

Some Basic OO Terms

- **Method:** Methods are functions inside a class. Like properties, methods can also be accessible by users.
- **Encapsulation:** Encapsulation is the mechanism that binds together code and the data it manipulates, and keeps both safe from outside interference and misuse.
  - Classes and objects (when created properly) protect the characteristics (properties) from direct access. This provides the object the opportunity to verify that any request to change a value in a property is valid before the change occurs.
- **Polymorphism:** Objects could be of any type. A discrete object can have discrete properties and methods which work separately to other objects. However a set of objects could be derived from a parent object and retain some properties of the parent class. This process is called polymorphism.

Some Basic OO Terms

- **Coupling:** Coupling is the behavior that indicates how classes are dependent on each other.
- **Inheritance:** The key process of deriving a new object by extending another object is called inheritance.
- Subclass: also called **Child** Class **or** derived class
  - When an object is derived from another object, the derived one is called the subclass of which it is derived from.
  - A class that inherits from another class.
- Super class: also called **Parent class or** base class
  - A class is super class to an object if that object is derived from it. To keep it simple, when you extend an object, the object which you are extending is the super class of a newly extended object.
  - A class that is inherited from by another class.
- Instance: Whenever you create an object by calling its constructor, it will be called an instance.
- **Data Abstraction:** Any representation of data in which the implementation details are hidden (abstracted).

General Coding Conventions

- These conventions will help you to maintain your application at a large extent.
  - It will increase the maintainability of your code.
  - It will also help you to write efficient code by avoiding duplicity and redundant objects.
  - It will make your code much more readable.
- In a single php file, never write more than one class at a time
  - Out of the scope of that class not write any procedural code.
  - Save any class with a proper naming convention.
  - For example save the file Emailer class introduced as class.emailer.php.
- What benefits can you achieve using this naming convention?
  - Well, without going inside that file, you are now at least confirmed that this file contains a class named "Emailer".

General Coding Conventions

- Never mix the case in filenames.
  - It creates ugly application structure. Go ahead with all small letters.
- Like classes, save any
  - interface as interface.name.php,
  - Abstract class as abstract.name.php and
  - Final class as final.name.php.
- While writing the name of properties or class variables, follow the same convention.

Classes IN PHP

- Recall from slide 13
- Class is something which defines your object.
  - For example your class is Car And **your Honda car** is object of car class.
- Class represents all properties and behaviors of object.
  - For example your car class will define that car should have color, number of door and **your car** which is an object will have color green and 2 doors. Your car is object of class car.
- In terms of programming we can say your car object is an instance of the car class.
- you can declare a class in PHP with the keyword class.

```
Class YourClassName{
 }
```

Object IN PHP

- Classes are useless without objects.
- Object is an instance of your class.
  - If you have class then you need to create object of the class to solve your problem using class.
- You can create object of your class by using **new** keyword.
  $objClass = new YourClassName();
- Now in above code you are creating object of class myClass in variable $objClass.
  - You can create multiple object of your same class. Every object is different from other.
    $objClass1 = new YourClassName();
    $objClass2 = new YourClassName();

Constructor of Classes and Objects

- Constructor is nothing but a function defined in your php class.
- Constructor function automatically called when you will create object of the class.
  - As soon as you will write $object = new yourClass() your constructor function of the class will be executed.
- In php4 you can create constructor by creating function with same name of your class.
- in php5 you can also create constructor by defining magic function __construct.

```
PHP4 constructor
class myClass{
var $var1;
function myClass(){
$this->var1 = 3;
}
}
```

```
PHP5 & 7 constructor
class myClass{
pvar $var1;
//Constructor of the class
public function __construct()
{
$this->var1 = 3;
}
}
```

Best Practice of Classes and Objects

- Instead of assigning variable of the classes after creating object it is good if you use constructor.
- Use visibility as required.
- Do not make your variable and method either more secure or completely open.
  - Over security will effect your flexibility, under security will distrust your structure.
- Follow same convention in your classes and objects.
  - Like start all public method with camel case, all protected method and variable prefix with _ etc. It will give you better visibility.
  - Do not try to do every thing in single class.
- Create class very specific to your requirement.
- Always try to create every class in separate file and follow same naming convention.

Visibility in PHP Classes

- **Public**: Public method or variable can be accessible from anywhere.
  - I mean from inside the class, out side the class and in child.
- **Private**: Method or property with private visibility can only be accessible inside the class.
  - You can not access private method or variable from outside of your class.
- **Protected**: Method or variable with protected visibility can only be access in the derived class. Or in other word in child class.
  - Protected will be used in the process of inheritance.

Inheritance in PHP

- Inheritance is nothing but a design principle in oop.
- By implementing inheritance you can inherit(or get) all properties and methods of one class to another class.
- The class who inherit feature of another class known as **child class or subclass**.
- The class which is being inherited is know as **parent class or super class**.
- Concept of the inheritance in oop is same as inheritance in real world.
  - For example, child inherits characteristics of their parent.
- Same is here in oop. One class is inheriting characteristics of another class.

Abstract classes in PHP

- As from name it seem like something that is hidden.
  - Yes nature of the abstract classes are same.
- Abstract classes are those classes which can not be directly initialized.
  - Or in other word you can not create object of abstract classes.
- Abstract classes always created for inheritance purpose.
- You can create abstract classes in php using **abstract** keyword.

```
abstract class MyAbstractClass{
 }
```

- Once you will make any class abstract in php you can not create object of that class.

Interface in PHP

- Interface in oop enforce definition of some set of method in the class.
- By implementing interface you are forcing any class to must declaring some specific set of methods in oop.
  - For example if you are creating class to render HTML element then it is necessary to set id and name of your html tag. So in this case you will create interface for that class and define method like setID and setName.
- You can create interface in php using keyword interface.
- By implementation of interface in php class you are specifying set of the method which classes must implement.

```
//interface for your class
interface yourInterface {
 public function functionName($value='');


}
```

```
//class that implement interface
class yourClass implements yourInterface
{
     function __construct($argument) { }
 public function functionName($value='')
 {
    return $value; } }
```

Differences between abstract class and interface in PHP

- It is not necessary that every method should be abstract in abstract class.
- But in interface every method is abstract.
- Multiple and multilevel both type of inheritance is possible in interface.
- But single and multilevel inheritance is possible in abstract classes.
- Method in interface must be public only.
- Method in abstract class could be public or protected both.
- In abstract class you can define as well as declare methods.
- But in interface you can only define your methods.

Static Properties

- Static properties of class is a property which is directly accessible from class with the help of ::(scope resolution operator).
- You can declare static property using **static** keyword.

```
class test
   {
   public static $a;//Static variable
   }
      test::$a = 5;
      echo test::$a;
```

Static property advantages

- Static variable or property are the best way to preserver value of the variable within the context of different instance.
- creating static variable or property is very useful if you want to share some data between the different object of the same class.

```
class test
    {
    private static $no_of_call = 0;
    public function __construct(){
    self::$no_of_call = self::$no_of_call + 1;
    echo "No of time object of the class created is: ". self::$no_of_call;
    }
    }
    $obj1 = new test(); // Prints No of time object of the class created is 1
    $objT2 = new test(); //Prints No of time object of the class created is 2
```

Static Methods or functions

- Same as **Static Properties**, **Methods in PHP** can be created function or method static using **static** keyword.
- You can access all visible static methods using :: like in static variables.

```
class test
    {
    static function abc($param1 , $param2)
    {
    echo "$param1 , $param2";
    }
    }
    test::abc("ankur" , "techflirt");
```

Static Methods and Property in Inheritance in PHP

- within the class static property can be accessed using **self** keyword.
- parent class property need to use parent keyword to access parent class property .

```
//Parent class
class parent
{

public static abc()

{



//your function body
}
```

```
// child class
class child extends parent
{
public static xyz()
{
//your function body
}
function callStatic()
{
self::xyz();
parent::abc();
}
}
```

Method Overriding in OOP

- In real word meaning of overriding phenomena of replacing the same parental behavior in child.
- Basic meaning of overriding in oop is same as real word meaning.
- In oop meaning of overriding is to replace parent class method in child class.
- In oop overriding is process by which you can re-declare your parent class method in child class.
- So basic meaning of overriding in oop is to change behavior of your parent class method.
- Overriding in PHP means you are free to change business logic, visibility and number of parameter.

Method Overriding in PHP

- In php Functions of parent class can be overrides by creation of same function with same name in the child class to override the function.

```
// Parent class
class testParent
{
// function one
public function f1()
{
echo 1;
}
Function two
public function f2()
{
echo 2;
}
}
```

```
// child class
class testChild extends testParent
{
      //overriding function f2

         function f2($a)
        {
           echo "$a";
         }
}

$a = new testChild();
$a->f2("ankur");
//it will print ankur
```

Final Keyword:

- PHP 5 introduces 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
// Base Class declared
class BaseClass {
      public function test() {
            echo "BaseClass::test() called<br>";
      }
      // Final method
      final public function moreTesting() {
            echo "BaseClass::moreTesting() called<br>"; }
      }
// ChildClass inhereted from base class
class ChildClass extends BaseClass {
      // final method overridden from base class
      public function moreTesting() {
            echo "ChildClass::moreTesting() called<br>";} }
?>
```

This example results in Fatal error:
Cannot override final method BaseClass::moreTesting()

Method Overloading in OOP

- In real word overloading means assigning extra work to same machine or person.
- Overloading in oop is same as overloading in real word.
- By process of method overloading you are asking your method to some extra work.
- Normally method overloading in oop is managed on the basis of the argument passed in function.
- We can achieve overloading in oop by providing different argument in same function.

Overloading in PHP

- Implementation of overloading need to take help of magic method in php.

```
<?php
class Overloader{
function __call($method, $arguments)
{
echo "You called a method named
{$method} with the following
arguments <br/>";
print_r($arguments);
echo "<br/>";
} }
$ol = new Overloader();
$ol->access(2,3,4);
$ol->notAnyMethod("boo");?>
```

```
You called a method named
access with the following
arguments
Array
(
[0] => 2
[1] => 3
[2] => 4
)
You called a method named
notAnyMethod with the
following arguments
Array
(
[0] => boo
)
```

Object Cloning in PHP

- If you will directly copy objects in php, then it will copy by reference, not by value.
- Means if you will change main object data then copied object will be affected.
- Also if you will change value of the copied object then main object value will be changed.
- So if you want to create copy of the **object which should never referenced to original object then you can take help of object cloning in php**.

Object copy or by reference copy

```
class test
    {
    public $a;
    private $b;
    function __construct($a, $b)
    {
    $this->a = $a;
    $this->b = $b;
    }
    }
    $a = new test("ankur" , "techflirt");
    $b = $a; //Copy of the object
    $a->a = "no Ankur";
    print_r($a);
    print_r($b);
```

Output

```
test Object
(
    [a] => no Ankur
    [b:test:private] => techflirt
)
test Object
(
    [a] => no Ankur
    [b:test:private] => techflirt
)
```

Implementation of Object Cloning in PHP

```
class test
    {
    public $a;
    private $b;
    function __construct($a, $b)
    {
    $this->a = $a;
    $this->b = $b;
    }
    }
    $a = new test("ankur" , "techflirt");
    $b = $a; //Copy of the object
    $c = clone $a; //clone of the object
    $a->a = "no Ankur";
    print_r($a);
    print_r($b);
    print_r($c);
```

output

```
test Object (
    [a] => no Ankur
    [b:test:private] => techflirt
)
test Object(
    [a] => no Ankur
    [b:test:private] => techflirt
)
```

```
test Object (
    [a] => ankur
    [b:test:private] =>
    techflirt
)
```

PHP Method Chaining

- With the help of **php method chaining or PHP function chaining** we can call more than one method or function of the class in single instruction.

```
$a = new Order();
$a->CreateOrder()->sendOrderEmail()->createShipment();
```

Or if you want to call it in multiline script then you can make same call like below example

```
$a = new Order();
$a->CreateOrder()
->sendOrderEmail()
->createShipment();
```

Magic Methods in PHP

- Magic methods in php are **some predefined function by php compiler which executes on some event**.
- Magic methods starts with prefix __, for example __call, __get, __set.
- __construct is a magic method which automatically call on creating object of the classes.
- __destruct: This magic method is called when object of your class is unset. This is just opposite of __construct.
- __get: This method called when your object attempt to read property or variable of the class which is inaccessible or unavailable.
- __set: This method called when object of your class attempts to set value of the property which is really inaccessible or unavailable in your class.

Magic Methods in PHP

- __isset: This magic methods trigger when isset() function is applied on any property of the class which is inaccessible or unavailable.
- __unset: is something opposite of isset method.
- __call: magic method trigger when you are attempting to call method or function of the class which is either inaccessible or unavailable.
- __callstatic: execute when inaccessible or unavailable method is in static context.
- __sleep methods trigger when you are going to serialize your class object.
- __wakeup executes when you are un serializing any class object.
- __toString executes when you are using echo on your object.
- __invoke called when you are using object of your class as function

__construct Method

- construct method trigger on creation of object. And __destruct triggers of deletion of object.

```
class test
{
function __construct()
{
echo 1;
}
function __destruct()
{
echo 2;
}
}
$objT = new test(); //__construct get automatically executed and print 1 on screen
unset($objT);//__destruct triggers and print 2.
```

__get __set __call and __callStatic

- These methods in php directly related with no accessible method and property of the class.
- **__get** takes one argument and executes when any inaccessible property of the method is called. It takes name of the property as argument.
- **__set** takes two property and executes when object try to set value in inaccessible property. It take first parameter as name of the property and second as the value which object is try to set.
- **__call** method fires when object of your class is trying to call method of property which is either non accessible or not available. It takes 2 parameter First parameter is string and is name of function. Second parameter is an array which is arguments passed in the function.
- **__callStatic** is a static magic method. It executes when any method of your class is called by static techniques.

Example

```
class test
    {
    function __get($name)
    {
    echo "__get executed with name $name ";
    }
    function __set($name , $value)
    {
    echo "__set executed with name $name , value $value";
    }
    function __call($name , $parameter)
    {
    $a = print_r($parameter , true); //taking recursive array
    in string
    echo "__call executed with name $name , parameter $a";
    }
```

Example …..

```
static function __callStatic($name , $parameter)
    {
    $a = print_r($parameter , true); //taking recursive array
    in string
    echo "__callStatic executed with name $name ,
    parameter $a";
    }
    }
    $a = new test();
    $a->abc = 3;//__set will executed
    $app = $a->pqr;//__get will triggerd
    $a->getMyName('ankur' , 'techflirt', 'etc');//__call willl be
    executed
    test::xyz('1' , 'qpc' , 'test');//__callstatic will be executed
```

__isset and __unset magic methods

- __isset and __unset magic methods in php are opposite of each other.
- **__isset** magic methods executes when function **isset()** is applied on property which is not available or not defined. It takes name of the parameter as an argument.
- **__unset** magic method triggers when unset() method is applied on the property which is either not defined or not accessible. It takes name of the parameter as an argument.

Example

```
class test
   {
   function __isset($name)
   {
   echo "__isset is called for $name";
   }
   function __unset($name)
   {
   echo "__unset is called for $name";
   }
   }
   $a = new test();
   isset($a->x);
   unset($a->c);
```

This is the end for this lecture