CHAPTER – 7

Object-Oriented Programming with PHP

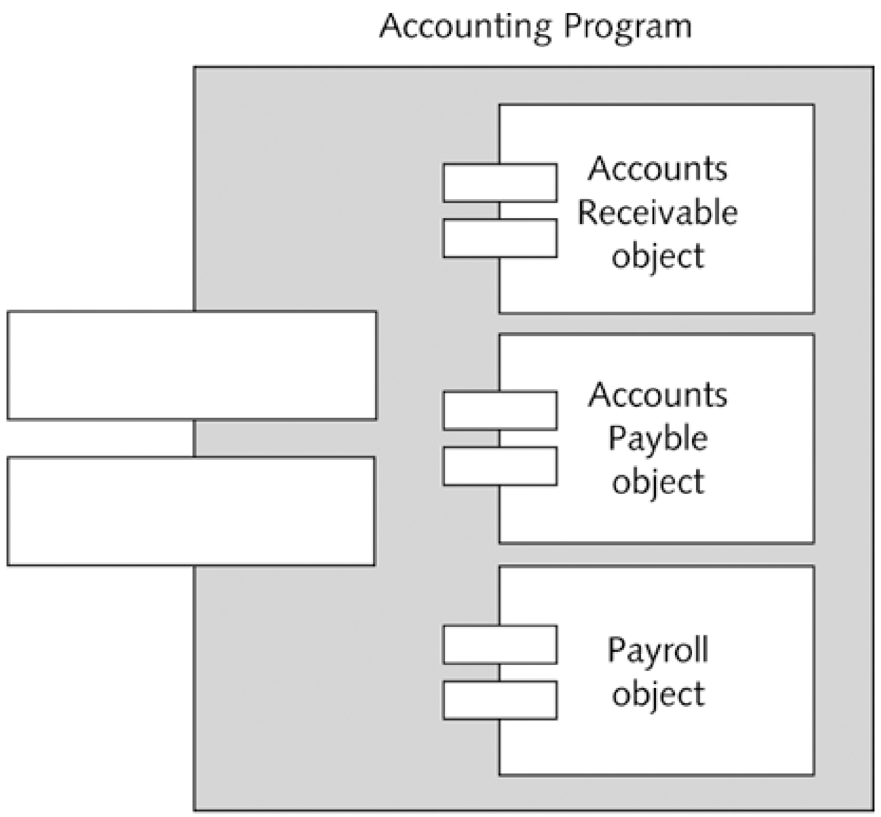
Object-Oriented Programming

- **Object-oriented programming** (OOP) refers to the creation of reusable software objects that can be easily incorporated into multiple programs
- An **object** refers to programming code and data that can be treated as an individual unit or component
- Objects are often also called **components**

Object-Oriented Programming

- **Data** refers to information contained within variables or other types of storage structures
- The functions associated with an object are called **methods**
- The variables that are associated with an object are called **properties** or **attributes**

Object-Oriented Programming



Understanding Encapsulation

- Objects are **encapsulated** – all code and required data are contained within the object itself
- Encapsulated objects hide all internal code and data
- An **interface** refers to the methods and properties that are required for a source program to communicate with an object

Understanding Encapsulation

- Encapsulated objects allow users to see only the methods and properties of the object that you allow them to see
- Encapsulation reduces the complexity of the code
- Encapsulation prevents other programmers from accidentally introducing a bug into a program, or stealing code

Object-Oriented Programming and Classes

- The code, methods, attributes, and other information that make up an object are organized into **classes**
- An **instance** is an object that has been created from an existing class
- Creating an object from an existing class is called **instantiating** the object
- An object **inherits** its methods and properties from a class — it takes on the characteristics of the class on which it is based

Using Objects in PHP Scripts

- Declare an object in PHP by using the **`new`** operator with a class constructor
- A **class constructor** is a special function with the same name as its class that is called automatically when an object from the class is instantiated
- The syntax for instantiating an object is:

```
$ObjectName = new ClassName();
```

USING OBJECTS IN PHP SCRIPTS

- The identifiers for an object name:
  - Must begin with a dollar sign
  - Can include numbers or an underscore
  - Cannot include spaces
  - Are case sensitive

    ```
    $Checking = new BankAccount();
    ```

  - Can pass arguments to many constructor functions

```
 $Checking = new BankAccount(01234587, 1021,
97.58);
```

Using Objects in PHP Scripts (continued)

- After an object is instantiated, use a hyphen and a greater-than symbol (->) to access the methods and properties contained in the object
- Together, these two characters are referred to as ***member selection notation***
- With member selection notation append one or more characters to an object, followed by the name of a method or property

Using Objects in PHP Scripts (continued)

- With methods, include a set of parentheses at the end of the method name, just as with functions
- Like functions, methods can also accept arguments

```
    $Checking->getBalance();
    $CheckNumber = 1022;
    $Checking-
>getCheckAmount($CheckNumber);
```

Defining Custom PHP Classes

- **Data structure** refers to a system for organizing data
- The functions and variables defined in a class are called **class members**
- Class variables are referred to as **data members** or **member variables**
- Class functions are referred to as **member functions** or **function members**

Defining Custom PHP Classes

- Classes:
  - Help make complex programs easier to manage
  - Hide information that users of a class do not need to access or know about
  - Make it easier to reuse code or distribute your code to others for use in their programs
- Inherited characteristics allow you to build new classes based on existing classes without having to rewrite the code contained in the existing one

Creating a Class Definition

- To create a class in PHP, use the `class` keyword to write a class definition
- A **class definition** contains the data members and member functions that make up the class
- The syntax for defining a class is:

```
class ClassName {
data member and member function definitions
}
```

Creating a Class Definition (continued)

- The ***ClassName*** portion of the class definition is the name of the new class
- Class names usually begin with an uppercase letter to distinguish them from other identifiers
- Within the class's curly braces, declare the data type and field names for each piece of information stored in the structure

```
class BankAccount {
data member and member function definitions
}
$Checking = new BankAccount();
```

Creating a Class Definition

- Class names in a class definition are not followed by parentheses, as are function names in a function definition

```
$Checking = new BankAccount();
echo 'The $Checking object is instantiated from the '
     . get_class($Checking) . " class.</p>";
```

- Use the `instanceof` operator to determine whether an object is instantiated from a given class

Some more information

- You can create object of the class in some different way also. Following is some of the example of creating object of class.
- E.g. $checking = ‘BankAccount’;
- $acc1 = new $checking(); (old style)
- $acc1=new BankAccount();
- Or $acc1 = new BankAccount;
- $acc2=new acc1; (new version style)

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.
- But from php5 you can also create constructor by defining magic function __construct

Playing with visibility and other feature of the constructor

- Reason behind creating constructor function public is it is accessible from outside of the class.
- This function is executed when we are creating object.
- So php will always through error if you will create your constructor private
- Constructor_private.php
- Conflict_in_constructor.php

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.

Best Practice of Classes and Objects

- Follow some 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. It will same your time and execution.
- Always try to create every class in separate file and follow some naming convention.

Available Visibility in PHP Classes

- There are 3 type of visibility available in php for controlling your property or method.
- **Public**: Public method or variable can be accessible from anywhere. I mean from inside the class, out side the class and in child class also.
- **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.

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.
- Refer doc file for the list.

Static Methods and Properties in PHP

- Static methods and properties in php can directly accessible without creating object of class.
- Your php class will be static class if your all methods and properties of the class is static.
- Static Methods and Properties in PHP will be treated as public if no visibility is defined.
- It is directly accessible from class with the help of ::(scope resolution operator)
- E.g. property_basic.php

Static Methods and Properties in PHP

- You can declare static property using **static** keyword.
- For within the class you can access static property using **self** keyword.
- If you are accessing parent class property then you need to use parent keyword.
- Static variable or property are the best way to preserve value of the variable within the context of different instance.
- E.g. self_parent.php

Static Methods or functions

- As in general class various process are same for methods and properties, **same is with Static Methods and Properties in PHP**.
- You can create your function or method static using **static** keyword.
- You can access all visible static methods for you using **::** like in static variables.

Inheritance in PHP

- Inheritance in php is introduced from php5 version.
- With the help of inheritance we can get all property and method of one class in another class.
- The class who inherit feature of another class known as **child class**. The class which is being inherited is know as **parent class**.
- E.g. basic_example.php

Multilevel and Multiple inheritance in PHP

- In php multilevel inheritance is possible but multiple inheritance is not possible.
- But hierarchical inheritance is possible in php.
- Hierarchical means Parent inherit property of grand parent class. Grand child inherit property of parent class.
- So in multilevel inheritance child can get some property of from grand parent class also.

Static Methods and Property in Inheritance

- We can use **$this->** keyword to get all property and method of parent class.
- But if your parent or child method is static, then you can access static methods or properties using **self** and **parent** keyword.
- Also this is not necessary to make method static if you want to use self or parent keyword.
- This is very useful if your parent and child both method is having property or method with same name.
- If both classes having same property and you want to call specific property or method then you can use this keyword.

Abstract Classes

- Abstract classes are those classes which can not be directly initialized and that you can not create object of it.
- If your class has at least one method abstract than your class is abstract class.
- **Abstract method is only declared but not defined**. This is not true definition as per my assumption.
- You can create abstract classes in php using **abstract** keyword.

Implementation of abstract method

- If you have an abstract method in your abstract class then once you inherit your abstract class then it is necessary to define your abstract method.
- You can define your abstract method in child class with the same visibility or less restricted visibility.

What is 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.

IMPLEMENTS

- To implement an interface, the *implements* operator is used.
- All methods in the interface must be implemented within a class;
- Classes may implement more than one interface if desired by separating each interface with a comma.
- It's possible for interfaces to have constants.
- Interface constants works exactly like class constants except they cannot be overridden by a class/interface that inherits them.
- You can not implement two interfaces with same function names.
- If you are using default argument then you can change your value of the argument.

Differences between abstract class and interface

- In abstract classes this is not necessary that every method should be abstract. 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 of php interface must be public only. Method in abstract class in php 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.

Overloading and Overriding in PHP

- Overriding is a process by which you can re-declare your parent class method in child class.
- Overriding is required when your parent class has some method, but in your child class you want it with different behavior.
- Overloading means assigning extra work to the existing method.
- We can not implement overloading by creating two functions with same names in class.
- So to implement overloading in php we will take help of magic method __call.
- Magic method __call invoked when method called by class object is not available in class.

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 CLONING WITH MAGIC METHOD __CLONE

- Suppose you want to change value of your property a of the test class in case of cloning of object in php.
- We can change behavior of the clone object in php using magic method __clone.
- Magic method clone is executed when object cloning is performed. As soon as php execute statement $c = clone $a, _ clone method invoked.

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.