CSE417: WEB ENGINEERING

Daffodil International University

Object oriented programming in PHP

- PHP, like most modern programming languages (C++, Java, Perl, JavaScript, etc.), supports the creation of objects.
- Creating an object requires you to first define an object class (containing variables and/or function definitions) and then using the “new” keyword to create an instance of the object class. (Note that the object must be defined before you instantiate it.)

Defining (declaring) a class

- Use the “class” keyword which includes the class name (case-insensitive, but otherwise following the rules for PHP identifiers). Note: The name “stdClass” is reserved for use by the PHP interpreter.

- Use the “$this” variable when accessing properties and functions of the current object. Inside a method this variable contains a reference to the object on which the method was called.

```
<?php

class student{
    public $id;
    public $name;

    public function info(){
        echo "id:".$this->id;
        echo "name:".$this->name;
    }
}
$ob1=new student();
$ob1->id=10;
$ob1->name="era";
$ob1->info();
?>
```

Constructors and destructors

- Constructors are methods that are (generally) used to initialize the object's properties with values as the object is created. Declare a constructor function in an object by writing a function with the name___construct().
- Destructors (defined with a function name of___destructor()) are called when an object is destroyed, such as when the last reference to an object is removed or the end of the script is reached (the usefulness of destructors in PHP is limited, since, for example dynamic memory allocation isn't possible in the same way that it is in C/C++).

Constructors and destructors

```
<?php

class student{
    public $id;
    public $name;
    public function __construct($id,$name){
        $this->id=$id;
        $this->name=$name;
    }
    public function info(){
        echo "id:".$this->id;
        echo "name:".$this->name;
    }
    public function __destruct(){
            // clean up resources or do something else
            echo "Destroying Object";
        }
}
$ob1=new student(10,"era");
$ob1->info();
?>
```

Inheritance

- Use the “extends” keyword in the class definition to define a new object that inherits from another.

```
<?php
class student{
    public $id;
    public $name;
    public function info(){
        echo "id:".$this->id;
        echo "name:".$this->name;
    }
}
class abc extends student{
    public $age;
       public function info(){
        echo "id:".$this->id;
        echo "name:".$this->name;
        echo "age:".$this->age;
    }
}
$ob1=new student;
$ob2=new abc;
$ob2->id=10;
$ob2->name="rahim";
$ob2->age=10;
$ob2->info();
?>
```

Get and Set method

- Used to access undefined variables

```
<?php
class MyClass
{
// set user's first name
public function setFName($fname)
{
$this->fname = $fname;
}
// get user's first name
public function getFName()
{
return $this->fname;
}

// set user's last name
public function setLName($lname)
{
$this->lname = $lname;
}

// get user's last name
public function getLName()
{
return $this->lname;
}

// set user's email address

public function setEmail($email)
{
$this->email = $email;

}
// get user's email address

public function getEmail()

{
return $this->email;

}
}
$user = new MyClass();
$user->setFName('Ajeet');
$user->setLName('Dubey');
$user->setEmail('adubey@gamil.com');

echo 'First Name: ' . $user-
>getFName().' </br>Last Name: ' .
$user->getLName() .'</br> Email: ' .
$user->getEmail();
?>
```

Exercise

- Design a form and validate its data.
- Design a user registration system
- **READINGS/Practice**
  - M Schafer: Ch. 29-32
  - https://www.w3schools.com/php/php_form_validation.asp
  - Designing a Sign-up/Log-in page