Desarrollo de Software VII

Laboratorio # 6 – Prof. Regis Rivera

Objetivo: Introducción a la programación orientada a objetos en PHP.

Laboratorio 6.1: Clase + Instancia de objeto en 1 solo archivo

Lab61.php

```
<?php

class cliente{

var $nombre;
var $numero;
var $peliculas_alquiladas;

function __construct($nombre,$numero){
$this->nombre=$nombre;
$this->numero=$numero;
$this->peliculas_alquiladas=array();
}

function __destruct(){
echo "<br>destruido: " . $this->nombre;
}

function dame_numero(){
return $this->numero;
}

}

//instanciamos un par de objetos cliente
$cliente1 = new cliente("Pepe", 1);
$cliente2 = new cliente("Roberto", 564);

//mostramos el numero de cada cliente creado
echo "<br> El identificador del cliente 1 es: " . $cliente1->dame_numero();
echo "<br> El identificador del cliente 2 es: " . $cliente2->dame_numero();

?>
```

Atributos

Constructor

Destructor

Método

Instancias de la clase (objetos)

Laboratorio 6.2: Clase + instancia de objeto en distintos archivos

```php
<?php

class cliente{
var $nombre;
var $numero;
var $peliculas_alquiladas;

function __construct($nombre,$numero){
$this->nombre=$nombre;
$this->numero=$numero;
$this->peliculas_alquiladas=array();
}

function __destruct(){
echo "<br>destruido: " . $this->nombre;
}

function dame_numero(){
return $this->numero;
}

}
?>
```

class_lib.php

```php
<?php

include("class_lib.php");

//instanciamos un par de objetos cliente
$cliente1 = new cliente("Pepe", 1);
$cliente2 = new cliente("Roberto", 564);

//mostramos el numero de cada cliente creado
echo "<br> El identificador del cliente 1 es: " . $cliente1->dame_numero();
echo "<br> El identificador del cliente 2 es: " . $cliente2->dame_numero();

?>
```

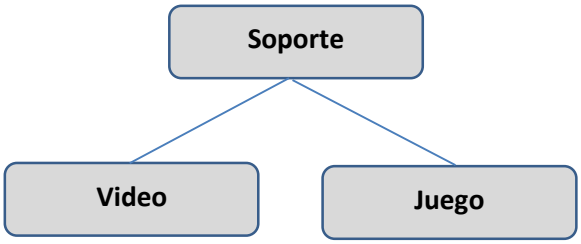
Lab62.php

Laboratorio 6.3: Herencia y modificadores de acceso

A continuación, se crearán 3 clases para poner en práctica el concepto de herencia de POO en PHP, estas serán agregadas al existente archivo **class_lib** (similar al lab62) este archivo tiene como objetivo albergar las distintas clases y se incluye en el archivo php que necesite instanciar sus objetos. Por temas de comodidad y estándar, es recomendable mantener las clases bajo el concepto de librería.

class_lib.php

Esquema de clases a diseñar:



Agregar las siguientes clases (al final del código antes de **?>**) en el existente archivo **class_lib.php:**

class_lib.php

```
class soporte{
public $titulo;
protected $numero;
private $precio;
function __construct($tit,$num,$precio){
$this->titulo = $tit;
$this->numero = $num;
$this->precio = $precio;
}
public function dame_precio_sin_itbm(){
return $this->precio;
}
public function dame_precio_con_itbm(){
return $this->precio * 0.07;
}
public function dame_numero_identificacion(){
return $this->numero;
}
public function imprime_caracteristicas(){
echo "<br>" . $this->titulo;
echo "<br>" . $this->precio . " (ITBM no incluido)";
}
}
```

class_lib.php

```
class video extends soporte {
protected $duracion;

function __construct($tit,$num,$precio,$duracion){
parent::__construct($tit,$num,$precio);
$this->duracion = $duracion;
}

public function imprime_caracteristicas(){
echo "<br> Película en Blu-Ray:";
parent::imprime_caracteristicas();
echo "<br>Duración: " . $this->duracion;
}
}
```

```
class juego extends soporte{
protected $consola; // consola del juego ej: DS Lite
protected $min_num_jugadores;
protected $max_num_jugadores;

Function __construct($tit,$num,$precio,$consola,$min_j,$max_j){
parent::__construct($tit,$num,$precio);
$this->consola = $consola;
$this->min_num_jugadores = $min_j;
$this->max_num_jugadores = $max_j;
}

public function imprime_caracteristicas(){
echo "<br>Juego para: " . $this->consola;
parent::imprime_caracteristicas();
echo "<br>" . $this->imprime_jugadores_posibles();
}

public function imprime_jugadores_posibles() {
if ($this->min_num_jugadores == $this->max_num_jugadores){
if ($this->min_num_jugadores==1)
echo "<br>Para un jugador";
else
echo "<br>Para " . $this->min_num_jugadores . " jugadores";
}
else {
echo "<br>De " . $this->min_num_jugadores . " a " . $this-
>max_num_jugadores . " jugadores.";
}
}
}
```

class_lib.php

Crear código PHP que instanciará los objetos de las clases creadas:

```
<?php

include("class_lib.php");

//Ejemplo de uso de la clase padre

$soporte1 = new soporte("The Soccer Game",22,3);
echo "<b>" . $soporte1->titulo . "</b>";
echo "<br>Precio: " . $soporte1->dame_precio_sin_itbm() . " dolares";
echo "<br>Precio ITBM incluido: " . $soporte1->dame_precio_con_itbm() . " dolares";

//ejemplo de uso de la clase hija 1

$mivideo = new video("Los Otros", 22, 4.5, "115 minutos");
echo "<br><br>";
echo "<b>" . $mivideo->titulo . "</b>";
echo "<br>Precio: " . $mivideo->dame_precio_sin_itbm(). " dolares";
```

Lab63.php

```
echo "<br>Precio ITBM incluido: " . $mivideo->dame_precio_con_itbm() . " dolares  ";
echo "<br>" . $mivideo->imprime_caracteristicas();


//Ejemplo de uso de la clase hija 2

$mijuego = new juego("Pes 18", 21, 2.5, "Xbox 360",1,2);
$mijuego->imprime_caracteristicas();
$mijuego->imprime_jugadores_posibles();
echo "<p>";
$mijuego2 = new juego("Fifa 18", 27, 3, "PS 4",1,2);
echo "<b>" . $mijuego2->titulo . "</b>";
$mijuego2->imprime_jugadores_posibles();

?>
```

Laboratorio 6.4: Métodos estáticos

```
class Foo {
public static $mi_static = 'foo';
public function staticValor() {
return self::$mi_static;
}
}

class Bar extends Foo {
public function fooStatic() {
return parent::$mi_static;
}
}
```

class_lib.php

```
<?php

include("class_lib.php");

print Foo::$mi_static . " value (1)<br>";

$foo = new Foo();
print $foo->staticValor() . " value (2)<br>";

print $foo->mi_static . "  value (3)<br>";
// "Propiedad" no definida mi_static
// $foo::mi_static no es posible

print Bar::$mi_static . "  value (4)<br>";

$bar = new Bar();
print $bar->fooStatic() . "  value (5)<br>";

?>
```

Lab64.php

Laboratorio 6.5: Método final

```
<?php

class ClaseBase {
public function test() {
echo "ClaseBase::test() llamada\n";
}
final public function masTests() {
echo "ClaseBase::masTests() llamada\n";
}
}

class ClaseHijo extends ClaseBase {
public function masTests() {
echo "ClaseHijo::masTests() llamada\n";
}
}

?>
```

Lab65.php

Explique la salida que da la aplicación

Laboratorio 6.6: Clase final

```
<?php

final class ClaseBase {
public function test() {
echo "ClaseBase::test() llamada\n";
}
// Aquí da igual si se declara el método como
// final o no
final public function moreTesting() {
echo "ClaseBase::moreTesting() llamada\n";
}
}

class ClaseHijo extends ClaseBase {
}

?>
```

Lab66.php

Explique la salida que da la aplicación