Programación IV

Laboratorio N°11

Prof. Oriel Cedeño

Objetivos del Laboratorio

- Comprender los conceptos básicos de la POO.
- Crear y utilizar clases y objetos en PHP.
- Implementar propiedades y métodos.
- Aplicar principios de herencia, polimorfismo, encapsulamiento y abstracción.

Primer Paso: Cree una carpeta llamada poo_lab dentro del directorio raíz de tu servidor web (por ejemplo, htdocs para XAMPP).

Definir una Clase: Abre coche.php y define una clase Coche con algunas propiedades y métodos básicos.

coche.php

```php
<?php
class Coche {
    public $color;
    public $marca;

    public function arrancar() {
        echo "El coche ha arrancado<br>";
    }

    public function detener() {
        echo "El coche se ha detenido<br>";
    }
}

// Crear un objeto de la clase Coche
$miCoche = new Coche();
$miCoche->color = "Rojo";
$miCoche->marca = "Toyota";

// Usar métodos del objeto
$miCoche->arrancar();
$miCoche->detener();

echo "Color: " . $miCoche->color . "<br>";
echo "Marca: " . $miCoche->marca . "<br>";
?>
```

Ejecutar el archivo: Abre tu navegador y ve a http://localhost/poo_lab/coche.php. Deberías ver el resultado de los métodos y propiedades del objeto.

Implementando Herencia

Crear una Clase Derivada: Añade una nueva clase CocheDeportivo que hereda de Coche.

```php
class CocheDeportivo extends Coche {
    public function turbo() {
        echo "El turbo está activado<br>";
    }
}

// Crear un objeto de la clase CocheDeportivo
$miCocheDeportivo = new CocheDeportivo();
$miCocheDeportivo->color = "Azul";
$miCocheDeportivo->marca = "Ferrari";

// Usar métodos del objeto
$miCocheDeportivo->arrancar();
$miCocheDeportivo->turbo();
$miCocheDeportivo->detener();

echo "Color: " . $miCocheDeportivo->color . "<br>";
echo "Marca: " . $miCocheDeportivo->marca . "<br>";
```

Ejecutar el archivo: Actualiza tu navegador en http://localhost/poo_lab/coche.php para ver el resultado.

Aplicar Polimorfismo

Modificar las Clases: Define métodos en ambas clases con el mismo nombre, pero comportamiento diferente.

```php
class Coche {
    public $color;
    public $marca;

    public function arrancar() {
        echo "El coche ha arrancado<br>";
    }

    public function detener() {
        echo "El coche se ha detenido<br>";
    }

    public function descripcion() {
        echo "Este es un coche normal<br>";
    }
}

class CocheDeportivo extends Coche {
    public function turbo() {
        echo "El turbo está activado<br>";
    }

    public function descripcion() {
        echo "Este es un coche deportivo<br>";
    }
}

$miCoche = new Coche();
$miCoche->descripcion();
```

```php
$miCocheDeportivo = new CocheDeportivo();
$miCocheDeportivo->descripcion();
```

Ejecutar el archivo: Actualiza tu navegador para ver el resultado del encapsulamiento.

Abstracción

Crear una Clase Abstracta: Define una clase abstracta Vehiculo y hereda de ella.

```php
abstract class Vehiculo {
    abstract protected function descripcion();
}

class Coche extends Vehiculo {
    public function descripcion() {
        echo "Este es un coche<br>";
    }
}

$miCoche = new Coche();
$miCoche->descripcion();
```

Ejecutar el archivo: Actualiza tu navegador para ver el resultado de la abstracción.

Constructores y Destructores

Implementar Constructores y Destructores: Define un constructor y un destructor en la clase Coche.

```php
class Coche {
    private $color;
    private $marca;

    public function __construct($color, $marca) {
        $this->color = $color;
        $this->marca = $marca;
        echo "Coche creado: $this->marca de color $this->color<br>";
    }

    public function __destruct() {
        echo "El coche $this->marca de color $this->color ha sido destruido<br>";
    }

    public function arrancar() {
        echo "El coche ha arrancado<br>";
    }

    public function detener() {
        echo "El coche se ha detenido<br>";
    }
}

$miCoche = new Coche("Amarillo", "Ford");
$miCoche->arrancar();
```

Ejecutar el archivo: Actualiza tu navegador para ver el resultado del constructor y el destructor.

Preguntas:

Clases y Objetos

- Define una clase y un objeto en tus propias palabras.
- ¿Qué propiedades y métodos tenía la clase Coche que creaste?
- ¿Cómo se crea un objeto en PHP? Proporciona un ejemplo.
- Describe un ejemplo de la vida real que podría ser representado por una clase y un objeto en programación.

Herencia

- ¿Qué es la herencia en la programación orientada a objetos?
- ¿Qué clase creó la clase CocheDeportivo y qué propiedades y métodos heredó de la clase Coche?
- Explica cómo la clase CocheDeportivo añadió funcionalidad adicional a la clase Coche.
- Proporciona un ejemplo de herencia en un contexto diferente al del coche (por ejemplo, animales, dispositivos electrónicos, etc.).

Polimorfismo

- Define polimorfismo en el contexto de la programación orientada a objetos.
- ¿Cómo demostraste el polimorfismo en la clase Coche y CocheDeportivo?

Encapsulamiento

- Explica qué es el encapsulamiento y por qué es importante en la programación orientada a objetos.
- ¿Cómo protegiste las propiedades color y marca en la clase Coche?
- ¿Qué son los métodos getter y setter y cómo se usan?

Abstracción

- Define abstracción en tus propias palabras.
- ¿Qué es una clase abstracta y cómo se utilizó en el archivo 05_abstraccion.php?

Constructores y Destructores

- ¿Qué es un constructor y cuándo se utiliza?
- ¿Cómo inicializaste las propiedades color y marca en el constructor de la clase Coche?
- ¿Qué es un destructor y cuándo se ejecuta?