Guía Práctica de Ejercicios Nº 8

En nuestra octava práctica continuaremos con la programación orientada a objetos, definiendo algunos conceptos fundamentales del paradigma POO, mencionando algunas de las características del mismo y luego pasaremos a la implementación de la POO en PHP versión 5.

Objetivo: Comprender y profundizar los conceptos básicos de la programación orientada a objetos, definiendo las clases y objetos con sus atributos y métodos. Definir, desarrollar y utilizar métodos setter, getter y toString. Entender y aplicar las relaciones entre clases utilizando como lenguaje de modelado UML y PHP para codificar las mismas. Entender el uso constantes de clase, su definición y uso, ya sea dentro de la definición de clase o fuera de la misma. Utilizar las palabras reservadas this y self para referenciar constantes y variables de una clase u objeto.

Introducción: En nuestra actualidad vemos que el desarrollo de Internet ha sido inminente y con ello las aplicaciones Web, por lo tanto, se hace indispensable el uso de un lenguaje que permita desarrollar aplicaciones Web como PHP, entre otros. Teniendo en cuenta la situación anterior es que veremos la como desarrollar aplicaciones Web que se ejecutarán del lado del servidor utilizando uno de los mejores lenguajes del ambiente del Software Libre. Teniendo en cuenta la magnitud del proyecto a desarrollar veremos la necesidad de emplear el paradigma orientado a objetos con PHP; que nos dará la posibilidad de diseñar, desarrollar y mantener el Software de manera profesional utilizando un paradigma antes mencionado.

Construcciones (palabras clave), variables globales y funciones que estudiaremos en esta unidad.

Construcciones del lenguaje y operadores con los que trabajaremos.

class nos permite crear una clase en PHP que es una colección de variables (atributos) y funciones (métodos) que trabajan con estas variables.

public nos permite definir atributos y métodos para que sean accedidos desde cualquier lugar.

private nos permite definir atributos y métodos para que sean accedidos desde la clase donde fueron definidos.

new nos permite instanciar un objeto de acuerdo a una clase.

this nos permite hacer referencia a una instancia (objeto) particular de la clase. Dejando la posibilidad de acceder a comportamientos y atributos de un objeto dentro de la definición de una clase.

self nos permite acceder a variables estáticas y constantes de una clase.

:: el operador de ámbito nos permite acceder a elementos estáticos, constantes y sobrescribir métodos de una clase.

Ejercicios

Utilizar los conceptos de programación Web y de base de datos

1) – Escriba los scripts PHP para definir en primer lugar una clase de nombre DB (**DB.php**) que implemente métodos para conectar, consultar, insertar, actualizar y borrar datos de bases de datos MySQL y PostgreSQL. Para lo cual deberá utilizar dos clases más (**Mysql.php** y **Postgresql.php**) que implementen los mismos métodos. En este ejercicio se le pedirá que utilice una clase Index (**index.php**) para controlar el acceso a las bases de datos.

Index.php

Métodos

(público) ejecutar()

DB.php

Atributos

(privado) _motor
(privado) _enlace

Métodos

(público) __construct($motor)
(público) conectar($consulta)
(público) consultar($consulta)
(público) insertar($consulta)
(público) actualizar($consulta)
(público) borrar($consulta)

Mysql.php

Atributos

(privado) _host
(privado) _username
(privado) _password
(privado) _dbname

Métodos

(público) __construct()
(público) conectarMysql()
(público) consultarMysql($consulta)
(público) insertarMysql($consulta)
(público) actualizarMysql($consulta)
(público) borrarMysql($consulta)

Postgresql.php

Atributos

(privado) _host
(privado) _username
(privado) _password
(privado) _dbname

Métodos

(público) __construct()
(público) conectarPgsql()
(público) consultarPgsql($consulta)
(público) insertarPgsql($consulta)
(público) actualizarPgsql($consulta)
(público) borrarPgsql($consulta)

index.php

```
<?php
require_once 'DB.php';
class Index
{
	public function ejecutar()
	{
		$oDb = new DB('MYSQL');

		$oDb->conectar();
		$resultados = $oDb->consultar('SELECT * FROM persona');
		$res = $oDb->insertar("INSERT INTO persona (id, apellidos, nombres, email)
VALUES (1,'Sbarbaro','Mario','mmsbarbaro@gmail.com')");
		$resultados = $oDb->consultar('SELECT * FROM persona');
	}
}
$oIndex = new Index();
$oIndex->ejecutar();
```

DB.php

```
<?php
require_once 'Mysql.php';
require_once 'Postgresql.php';
class DB
{
	private $_motor;
	private $_enlace;

	public function __construct($motor)
	{
		$this->_motor = $motor;
	}
	public function conectar()
	{
		if ($this->_motor == 'MYSQL') {
			$my = new Mysql();
			$this->_enlace = $my->conectarMysql();
		} else if ($this->_motor == 'POSTGRESQL') {
			$pg = new Postgresql();
```

```
                $this->_enlace = $pg->conectarPgsql();
        }
}
public function consultar($consulta)
{
        if ($this->_motor == 'MYSQL') {
                $my = new Mysql();
                $resultado = $my->consultarMysql($consulta, $this->_enlace);
        } else if ($this->_motor == 'POSTGRESQL') {
                $pg = new Postgresql();
                $resultado = $pg->consultarPgsql($consulta, $this->_enlace);
        }

        while ($fila=mysql_fetch_array($resultado)) {
                $aResultado[] = $fila;
        }

        return $aResultado;
}
public function insertar($consulta)
{
        if ($this->_motor == 'MYSQL') {
                $my = new Mysql();
                $resultado = $my->insertarMysql($consulta, $this->_enlace);
        } else if ($this->_motor == 'POSTGRESQL') {
                $pg = new Postgresql();
                $resultado = $pg->insertarPgsql($consulta, $this->_enlace);
        }
        if ($resultado == true) {
                        return true;
                } else if ($resultado == false) {
                        return false;
        }
}
public function actualizar($consulta)
{
        if ($this->_motor == 'MYSQL') {
                $my = new Mysql();
                $resultado = $my->actualizarMysql($consulta, $this->_enlace);
        } else if ($this->_motor == 'POSTGRESQL') {
                $pg = new Postgresql();
                $resultado = $pg->actualizarPgsql($consulta, $this->_enlace);
        }
        if ($resultado == true) {
                        return true;
                } else if ($resultado == false) {
                        return false;
        }
}
public function borrar($consulta)
{
        if ($this->_motor == 'MYSQL') {
```

```
                        $my = new Mysql();
                        $resultado = $my->borrarMysql($consulta, $this->_enlace);
                } else if ($this->_motor == 'POSTGRESQL') {
                        $pg = new Postgresql();
                        $resultado = $pg->borrarPgsql($consulta, $this->_enlace);
                }
                if ($resultado == true) {
                                return true;
                        } else if ($resultado == false) {
                                return false;
                }
        }
}
```

Mysql.php

```
<?php
class Mysql
{
        private $_host;
        private $_username;
        private $_password;
        private $_dbname;

        public function __construct()
        {
                $this->_host = '127.0.0.1';
                $this->_username = 'root';
                $this->_password = '';
                $this->_dbname = 'prueba';
        }
        public function conectarMysql()
        {
                $enlace = mysql_connect($this->_host, $this->_username, $this->_password);

                return $enlace;
        }
        public function consultarMysql($consulta, $enlace)
        {
                mysql_select_db($this->_base, $enlace);
                $resultado = mysql_query($consulta, $enlace);

                return $resultado;
        }
        public function insertarMysql($consulta, $enlace)
        {
                return $this->consultarMysql($consulta, $enlace);
        }
        public function actualizarMysql($consulta, $enlace)
        {
                return $this->consultarMysql($consulta, $enlace);
        }
        public function borrarMysql($consulta, $enlace)
```

```
	{
		return $this->consultarMysql($consulta, $enlace);
	}

}
```

Postgresql.php

```
<?php
class Postgresql
{
	private $_host;
	private $_username;
	private $_password;
	private $_dbname;

	public function __construct()
	{
		$this->_host = '127.0.0.1';
		$this->_username = 'postgresql';
		$this->_password = '';
		$this->_dbname = 'prueba';
	}
	public function conectarPgsql()
	{
		return pg_connect("host=".$this->_host." port=5432 dbname=".$this->_dbname."
user=".$this->_username." password=".$this->_password);
	}
	public function consularPqsql($consulta, $enlace)
	{
		return pg_query($consulta, $enlace);
	}
	public function insertarPgsql($consulta, $enlace)
	{
		return $this->consultarPgsql($consulta, $enlace);
	}
	public function actualizarPgsql($consulta, $enlace)
	{
		return $this->consultarPgsql($consulta, $enlace);
	}
	public function borrarPgsql($consulta, $enlace)
	{
		return $this->consultarPgsql($consulta, $enlace);
	}
}
```

2) – Teniendo en cuenta el ejemplo de clases y objetos de PHP (Index, Facultad, Persona, Mascota) de la clase de teoría se le solicitará que termine de implementar el mismo, creando y definiendo las clases que corresponde.

index.php

```
<?php
require_once 'Facultad.php';
require_once 'Mascota.php';
```

```
require_once 'Persona.php';
class Index
{
	public function ejecutar()
	{
		$oPersona = new Persona('Garcia', 'Juan', 23);
		$oPerro = new Mascota('Kirby', 'blanco, negro y marron');

		echo $oPersona->tocar($oPerro, 'cabeza');
	}
}
$oIndex = new Index();
$oIndex->ejecutar();
```

Persona.php

```
<?php
require_once 'Mascota.php';
class Persona
{
	private $_apellidos;
	private $_nombres;
	private $_edad;
	private $_hermanos = array();
	private $_mascotas = array();
	public function __construct($apellidos, $nombres, $edad)
	{
		$this->_apellidos = $apellidos;
		$this->_nombres = $nombres;
		$this->_edad = $edad;
	}
	public function agregarHermano(Persona $persona)
	{
		$this->_hermanos[] = $persona;
	}
	public function agregarMastoca(Mascota $mascota)
	{
		$this->_mascotas[] = $mascota;
	}
	public function tocar(Mascota $mascota, $lugar)
	{
		return $mascota->tocar($lugar);
	}
}
```

Mascota.php

```
<?php
class Mascota
{
	private $_nombre;
	private $_color;
	public function __construct($nombre, $color)
	{
```

```
                $this->_nombre = $nombre;
                $this->_color = $color;
        }
        public function tocar($lugar)
        {
                if ($lugar = 'cabeza') {
                        return $this->_moverCola();
                }
        }
        private function _moverCola()
        {
                return 'muevo la cola!!!';
        }

}
```

Facultad.php

```
<?php
require_once 'Persona.php';
class Facultad
{
        private $_nombre;
        private $_alumnos;
        public function __construct($nombre)
        {
                $this->_nombre = $nombre;
        }
        public function agregarAlumno(Persona $persona)
        {
                $this->_alumnos[] = $persona;
        }
}
```

3) – Realice un script de nombre Persona.php que defina y use una clase (Persona) con los siguientes atributos y métodos:

Atributos

- (privado) tipoDocumento
- (privado) numeroDocumento
- (privado) apellidos
- (privado) nombres
- (privado) telefono
- (privado) correoElectronico

Métodos

- (público) __construct()
- (público) cambiarNumeroTelefono()
- (público) cambiarCorreoElectronico()
- (público) obtenerApellidos()
- (público) obtenerNombres()

index.php

```
<?php

require_once 'Persona.php';
```

```
class Index
{
	public function ejecutar()
	{
		$oPersona = new Persona('DNI', 22444222, 'Garcia', 'Juan');
		echo $oPersona->obtenerApellidos();
		echo “<br />”;
		echo $oPersona->obtenerNombres();
	}
}

$oIndex = new Index();
$oIndex->ejecutar();
```

Persona.php

```
<?php
class Persona
{
	private $_tipoDocumento;
	private $_numeroDocumento;
	private $_apellidos;
	private $_nombres;
	private $_telefono;
	private $_correoElectronico;
	public function __construct($tipoDocu, $numDocu, $apellidos, $nombres)
	{
		$this->_tipoDocumento = $tipoDocu;
		$this->_numeroDocumento = $numDocu;
		$this->_apellidos = $apellidos;
		$this->_nombres = $nombres;
	}
	private function _establecerTelefono($telefono)
	{
		$this->_telefono = $telefono;
	}
	public function cambiarTelefono($telefono)
	{
		$this->_establecerTelefono($telefono);
	}
	private function establecerCorreoElectronico($email)
	{
		$this->_correoElectronico = $email;
	}
	public function cambiarCorreoElectronico($email)
	{
		$this->establecerCorreoElectronico($email);
	}
	public function obtenerApellidos()
	{
		return $this->_apellidos;
	}
```

```
	public function obtenerNombres()
	{
		return $this->_nombres;
	}
}
```

4) – Mejore el script que accede a la base de datos del primer ejercicio de practica empleando constantes para determinar el motor seleccionado en la construcción de un objeto de la clase DB (DB.php).

Index.php

```
<?php
require 'DB.php';

class Index
{
	public function ejecutar()
	{
		$oDbPgsql = new DB(DB::PGSQL);

		//más instrucciones para acceder a la DB en Postgresql
	}
}
$oIndex = new Index();
$oIndex->ejecutar();
```

DB.php

```
<?php
require_once 'Mysql.php';
require_once 'Postgresql.php';
class DB
{
	const MYSQL = 1;
	const PGSQL = 2;

	private $_motor;
	private $_enlace;

	public function __construct($motor = self::MYSQL)
	{
		$this->_motor = $motor;
	}
	public function conectar()
	{
		if ($this->_motor == self::MYSQL) {
			$my = new Mysql();
			$this->_enlace = $my->conectarMysql();
		} else if ($this->_motor == self::POSTGRESQL) {
			$pg = new Postgresql();
			$this->_enlace = $pg->conectarPgsql();
		}
	}
	public function consultar($consulta)
```

```
{
        if ($this->_motor == self::MYSQL) {
                $my = new Mysql();
                $resultado = $my->consultarMysql($consulta, $this->_enlace);
        } else if ($this->_motor == self::POSTGRESQL) {
                $pg = new Postgresql();
                $resultado = $pg->consultarPgsql($consulta, $this->_enlace);
        }

        while ($fila=mysql_fetch_array($resultado)) {
                $aResultado[] = $fila;
        }

        return $aResultado;
}
public function insertar($consulta)
{
        if ($this->_motor == self::MYSQL) {
                $my = new Mysql();
                $resultado = $my->insertarMysql($consulta, $this->_enlace);
        } else if ($this->_motor == self::POSTGRESQL) {
                $pg = new Postgresql();
                $resultado = $pg->insertarPgsql($consulta, $this->_enlace);
        }
        if ($resultado == true) {
                        return true;
                } else if ($resultado == false) {
                        return false;
        }
}
public function actualizar($consulta)
{
        if ($this->_motor == self::MYSQL) {
                $my = new Mysql();
                $resultado = $my->actualizarMysql($consulta, $this->_enlace);
        } else if ($this->_motor == self::POSTGRESQL) {
                $pg = new Postgresql();
                $resultado = $pg->actualizarPgsql($consulta, $this->_enlace);
        }
        if ($resultado == true) {
                        return true;
                } else if ($resultado == false) {
                        return false;
        }
}
public function borrar($consulta)
{
        if ($this->_motor == self::MYSQL) {
                $my = new Mysql();
                $resultado = $my->borrarMysql($consulta, $this->_enlace);
        } else if ($this->_motor == self::POSTGRESQL) {
                $pg = new Postgresql();
```

```
                $resultado = $pg->borrarPgsql($consulta, $this->_enlace);
            }
            if ($resultado == true) {
                return true;
            } else if ($resultado == false) {
                return false;
            }
        }
}
```

5) – Para los puntos 2 y 3 agregar el método __toString() mostrando información que identifique el objeto.

DB.php

```
public function __toString()
{
    return “Conectado a un motor:  ”.$this->_motor;
}
```

Mysql.php

```
public function __toString()
{
    return “Servidor MySQL conectado al: ”.$this->_servidor;
}
```

Postgresql.php

```
public function __toString()
{
    return “Servidor PostgreSQL conectado al: ”.$this->_servidor;
}
```

Facultad.php

```
public function __toString()
{
    return $this->_nombre;
}
```

Persona.php

```
public function __toString()
{
    return $this->_apellidos.”, “.$this->_nombres;
}
```

Mascota.php

```
public function __toString()
{
    return $this->_nombre;
}
```

Persona.php

```
public fucntion __toString()
{
    return $this->_tipoDocumento. “ “.$this->_numeroDocumento;
}
```

Anotaciones de los Ejercicios