Curso de **Desarrollo Web** en **PHP orientado a objetos** con **MVC**

Eugenia Bahit, Abril 2015

5

Diseño de sistemas ABM

Introducción a los sistemas ABM

1 Preparar esqueleto del archivo MVC

1) Crear archivo .php (dentro de www) con el nombre de aquello que se desea gestionar
   Ejemplo: **`producto.php`**

2) Definir las clases *Model, *View y *Controller

```
<?php

class ProductoModel { }

class ProductoView { }

class ProductoController { }

?>
```

3) Definir las propiedades (datos necesarios) del modelo

```
class ProductoModel {
    public $producto_id = 0;
    public $denominacion = '';
    public $precio = 0.0;
    public $detalles = '';
}
```

2 Crear tabla en la base de datos

1) Crear archivo .sql (dentro de www/sql) con el mismo nombre que el archivo anterior
Ejemplo: **`producto.sql`**

2) Escribir la sentencia SQL para crear la tabla:

```
CREATE TABLE producto (
    producto_id INT(11) NOT NULL AUTO_INCREMENT PRIMARY KEY
    , denominacion VARCHAR(100)
    , precio DECIMAL(6, 2)
    , detalles BLOB
) ENGINE=InnoDB;
```

3) Ejecutar el archivo SQL:

```
:~$ mysql -u root -p mirpoyectodb < www/sql/producto.sql
```

3 Preparar el modelo

```
class ProductoModel {

    public $producto_id = 0;
    public $denominacion = '';
    public $precio = 0.0;
    public $detalles = '';

    function insert() { }

    function update() { }

    function select() { }

    function delete() { }
}
```

- El modelo es el único que se conecta a la base de datos
- Los 4 métodos del modelo, corresponden a los 4 tipos de consultas SQL básicas: INSERT, UPDATE, SELECT y DELETE

4 Preparar la vista

```
class ProductoView {

    function agregar() {
        print 'Mostrar formulario de Alta';
    }

    function editar() {
        print 'Mostrar formulario de Edición';
    }

    function ver() {
        print 'Mostrar productos';
    }

}
```

- La vista solo contará con funciones que se encarguen de mostrar algo al usuario

5 Preparar el controlador

```
class ProductoController {

	function __construct() {
		$this->view = new ProductoView();
		$this->model = new ProductoModel();
	}

	function agregar() {
		$this->view->agregar();
	}

	function guardar() {
		$this->model->insert();
		header('Location: /productos/ver');
	}

	function editar() {
		$this->view->editar();
	}

	function actualizar() {
		$this->model->update();
		header('Location: /productos/ver');
	}

	function eliminar() {
		$this->model->delete();
		header('Location: /productos/ver');
	}

	function ver() {
		$this->view->ver();
	}
}
```

- El controlador conecta la vista con el modelo
- Las funciones se llaman “recursos”
- Se accede a los recursos desde la URL: **/archivo/recurso**
  Por ejemplo: /producto/agregar
- Los recursos que solo conectan con el modelo, finalmente redirigen al usuario a un recurso que conecte con la vista

El **controlador** será luego, el encargado de **sanear y validar los datos** provistos por el usuario

6 Modificar el controlador de la aplicación

```
<?php

$peticion = $_SERVER['REQUEST_URI'];
list($null, $archivo, $recurso) = explode('/', $peticion);

$controller_name = ucwords($archivo) . 'Controller';
require_once "/$archivo.php";

$controller = new $controller_name;
$controller->$recurso();

?>
```

Probar ahora, ingresar en:
http://miproyecto.local/producto/agregar
http://miproyecto.local/producto/editar
http://miproyecto.local/producto/ver
http://miproyecto.local/producto/guardar ← redireccionará /producto/ver