

IES NTRA. SRA. DE LOS REMEDIOS DE UBRIQUE

Tema 06 – Modelo Vista Controlador

2º DAW – Desarrollo Web Entorno Servidor

Profesor Juan Carlos Moreno

CURSO 2023/2024

Tabla de Contenido

6	Modelo Vista Controlador.....	3
6.1	Introducción.....	3
6.2	MVC	3
6.3	Estructura Básica Proyecto MVC	4
6.3.1	Index.php	4
6.3.2	Archivo .htaccess	5
6.3.3	Carpetas Views, Models y Controllers	5

6 Modelo Vista Controlador

6.1 Introducción

El **Modelo Vista Controlador (MVC)** es un patrón o metodología de arquitectura de software que implementa nuestro proyectos a tres capas:

- Interfaz de usuario
- Reglas de negocio
- Acceso a los datos.

Este patrón de arquitectura de software se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento.

Se ha convertido en una arquitectura de diseño estándar que se utiliza tanto en componentes gráficos básicos hasta sistemas empresariales.

La mayoría de los frameworks modernos utilizan MVC (o alguna adaptación del MVC) para la arquitectura, entre ellos podemos mencionar a Ruby on Rails, Django, AngularJS, Laravel, etc..

Hay algo de **esfuerzo** necesario para aprender a utilizar un marco **MVC** en php. Sin embargo, para el desarrollador de aplicaciones Web grandes, este esfuerzo debe ser recompensado por los numerosos beneficios de utilizar un patrón de diseño MVC, tales como:

- Aplica la modularidad y la partición de aplicación.
- Aumenta la creación de roles específicos en el desarrollo.
- Aumenta la capacidad de gestión de código.
- Aumento de la extensibilidad del código (Capacidad de adaptación a cambios).

6.2 MVC

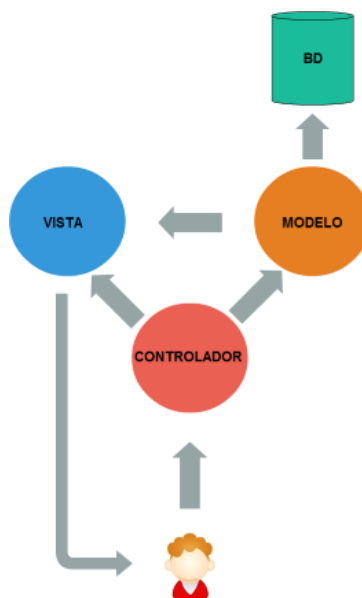


Ilustración 1. Model Vista Controlador

El patrón MVC nos indica que para estructurar correctamente nuestro desarrollo, debemos separar el código de nuestra aplicación en 3 partes diferenciadas:

- El **Modelo** se encarga de todo lo que tiene que ver con la persistencia de datos. Guarda y recupera la información del medio persistente que utilicemos, ya sea una base de datos, ficheros de texto, XML, etc.
- La **Vista** presenta la información obtenida con el modelo de manera que el usuario la pueda visualizar.
- El **Controlador**, dependiendo de la acción solicitada por el usuario, es el que pide al modelo la información necesaria e invoca a la plantilla (de la vista) que corresponda para que la información sea presentada.

6.3 Estructura Básica Proyecto MVC













Nombre	Fecha de modifica...	Tipo	Tamaño
 bd	10/02/2020 23:01	Carpeta de archivos	
 class	10/02/2020 23:09	Carpeta de archivos	
 config	10/02/2020 23:01	Carpeta de archivos	
 controllers	10/02/2020 23:09	Carpeta de archivos	
 imagenes	10/02/2020 23:08	Carpeta de archivos	
 libs	10/02/2020 23:01	Carpeta de archivos	
 models	10/02/2020 23:06	Carpeta de archivos	
 public	10/02/2020 23:01	Carpeta de archivos	
 template	10/02/2020 23:06	Carpeta de archivos	
 views	10/02/2020 23:10	Carpeta de archivos	
 .htaccess	02/05/2019 21:25	Archivo HTACCESS	1 KB
 index	06/05/2019 18:06	Archivo PHP	1 KB

Ilustración 2. Estructura Proyecto MVC

6.3.1 Index.php

El archivo más importante en un proyecto MVC es el **index.php**. Todas las peticiones URL que realice el usuario pasarán por este fichero. Toda acción que se ejecute en nuestra aplicación tendrá que llamar al index y este tendrá que cargar el controlador asociado a dicha acción, el modelo y la vista si procede.

Index.php

```
<?php

require_once 'libs/database.php';
require_once 'libs/controller.php';
require_once 'libs/model.php';
require_once 'libs/view.php';

require_once 'libs/app.php';
require_once 'config/config.php';
$app = new App();
```

?>

Nuestro archivo index.php simplemente cargaría los archivos correspondientes a la gestión del proyecto y crearía un objeto de la clase App que es la que se encargará de cargar el controlador asociado a la petición URL.

6.3.2 Archivo .htaccess [Convierte ruta en parametros](#)

Este archivo es fundamental para el enrutamiento de los distintos controladores y se ha de colocar en el directorio raíz de nuestro proyecto MVC junto con el index.php.

La función que tiene es devolver cada petición URL como un parámetro que puede ser extraído con \$_GET por nuestra app.

El contenido de dicho archivo ha de ser el siguiente

```
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-d
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-l
RewriteRule ^(.*)$ index.php?url=$1 [L,QSA]
```

6.3.3 Carpetas Views, Models y Controllers

Son las tres carpetas más importantes de nuestro proyecto y la que refleja el diseño a tres capas separando el acceso directo a la base de datos, el negociado y gestión de dichos datos y las salidas mediante las vistas.

6.3.3.1 Views




Nombre	Fecha de modifica...	Tipo	Tamaño
 ayuda	10/02/2020 23:01	Carpeta de archivos	
 error	10/02/2020 23:01	Carpeta de archivos	
 main	10/02/2020 23:01	Carpeta de archivos	

Ilustración 3. Views

Por cada vista que necesite para mi proyecto crearé una carpeta dentro de views, dentro de dicha carpeta crearé el archivo index.php con el código HTML, CSS, PHP, ... correspondiente a dicha vista.

Las tres vistas que necesito cuando inicio un proyecto son las siguientes:

- **Main.** Es la vista que se cargará cuando se realice la petición URL: index.php
- **Error.** Es la vista asociada que se cargará cuando se haga una petición URL que no tiene asociado ningún controlador.
- **Ayuda.** Es la vista asociada cada vez que necesite mostrar un mensaje de ayuda, aunque esta es opcional.

6.3.3.2 Models

Esta carpeta se almacenará los modelos de nuestro proyecto.

Los modelos contienen la conexión a la base de datos y los métodos asociados al controlador, prácticamente esta clase coincide con los archivos php que hemos diseñado para establecer la conexión y los métodos con nuestras bases de datos.

A parte de la conexión los modelos contienen los métodos típicos para realizar CRUD:

- INSERT
- UPDATE
- DELETE
- SELECT
- Resto métodos

Es muy importante la nomenclatura que voy a usar para nombrar al archivo php que contiene el modelo. El nombre del modelo ha de ir asociado siempre al nombre del controlador, por ejemplo si disponemos del controlador **articulos** el modelo el archivo que contendrá el modelo asociado se llamará **articulosModel.php**.



Nombre	Fecha de modifica...	Tipo	Tamaño
 articulo	27/01/2020 18:49	Archivo PHP	1 KB
 articulosModel	28/01/2020 11:18	Archivo PHP	6 KB

Ilustración 4. models

6.3.3.3 Controllers Lanzará error 404 si no se encuentra

En esta carpeta es donde ubicaremos los controladores. Hay que entender que cada petición URL en principio va asociada a un controlador, es el archivo app.php que veremos más adelante el que se encarga de realizar dicha labor.

El controlador es el archivo más importante de nuestro proyecto, ya que se encargará de cargar el modelo asociado si procede, realizar la gestión con esos datos y enviarlos a la vista.

Hay controladores que no necesitan acceder a ninguna base de datos, por lo tanto no tienen asociado ningún modelo, también puede ocurrir que haya controladores que lo único que realicen sean acceso y gestión de datos y no tengan asociada ninguna vista.




Nombre	Fecha de modifica...	Tipo	Tamaño
 ayuda	07/05/2019 18:16	Archivo PHP	1 KB
 error	07/05/2019 18:17	Archivo PHP	1 KB
 main	27/01/2020 22:03	Archivo PHP	1 KB

Ilustración 5. Controllers

En nuestra configuración básica dispondremos de 3 controladores:

- Main. Controlador asociado a la URL index.php
- Error. Controlador asociado a una URL no válida
- Ayuda. Solicitud de ayuda.