Modelo Vista Controlador (MVC)

Modelo Vista Controlador

¿Qué es?

El Modelo Vista Controlador (MVC) es una guía o patrón de arquitectura de software encargado de separar la lógica de negocio de la interfaz del usuario.

MVC no es nuevo y no se concibió originariamente para aplicaciones web, **pues fue descrito en 1979 por Trygve Reenskaug mientras trabajaba en Xerox PARC**. En definitiva, es una patrón multiplatorma.

Esta compuesto por 3 componentes y nos permite:

- Escalabilidad del sistema
- Mantenibilidad del código y aplicación.
- Funcionabilidad del sistema.

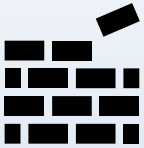
MVC

¿En qué nos ayuda?

Mediante el MVC podemos acércanos a los principios básicos de diseño del software. Que son unas buenas prácticas para la programación orientada a objetos.

- Reutilización de Código.
- Escalabilidad.
- Principios SOLID que nos ayudan a escribir software de calidad, mejorar la cohesión disminuyendo el acoplamiento y crear código que sea más fácil de leer, testear y mantener.

MVC



¿Qué es SOLID?

Son un patrón de ***buenas prácticas*** y no siempre se pueden cumplir todos a la vez, ya que es casi imposible en algunas ocasiones.

Si no se usan bien pueden aumentar la complejidad del código, hacerlo ambiguo y/o confuso.

Los principios SOLID son:

- **S** → Principio de Responsabilidad Única

  Clases Simples, únicamente hacen una cosa.

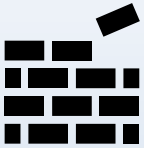
- **O** → Principio Open / Closed

  Clases abiertas a la extensión y cerradas a la modificación.

- **L** → Principio de Sustitución de Liskov

  Si usamos una clase, y esta clase es extendida, tenemos que poder utilizar cualquiera de las clases hijas y que el programa siga siendo válido.

MVC



¿Qué es SOLID?

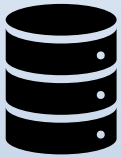
- **I** → Principio de Segregación de Interfaces

  Ninguna clase debería depender de métodos que no usa (fat interfaces)
- **D** → Principio de Inversión de Dependencias
  - Las clases de alto nivel no deberían depender de las clases de bajo nivel. Ambas deberían depender de las abstracciones.
  - Las abstracciones no deberían depender de los detalles. Los detalles deberían depender de las abstracciones

MVC

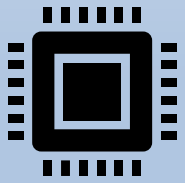
Componentes

Consta de 3 componentes (MVC):



- Modelo:
  - Se encarga de los datos, generalmente mediante la consulta de una base de datos.
  - Acceder directamente a los datos mediante consultas, actualizando o eliminando los mismos como intermediario.

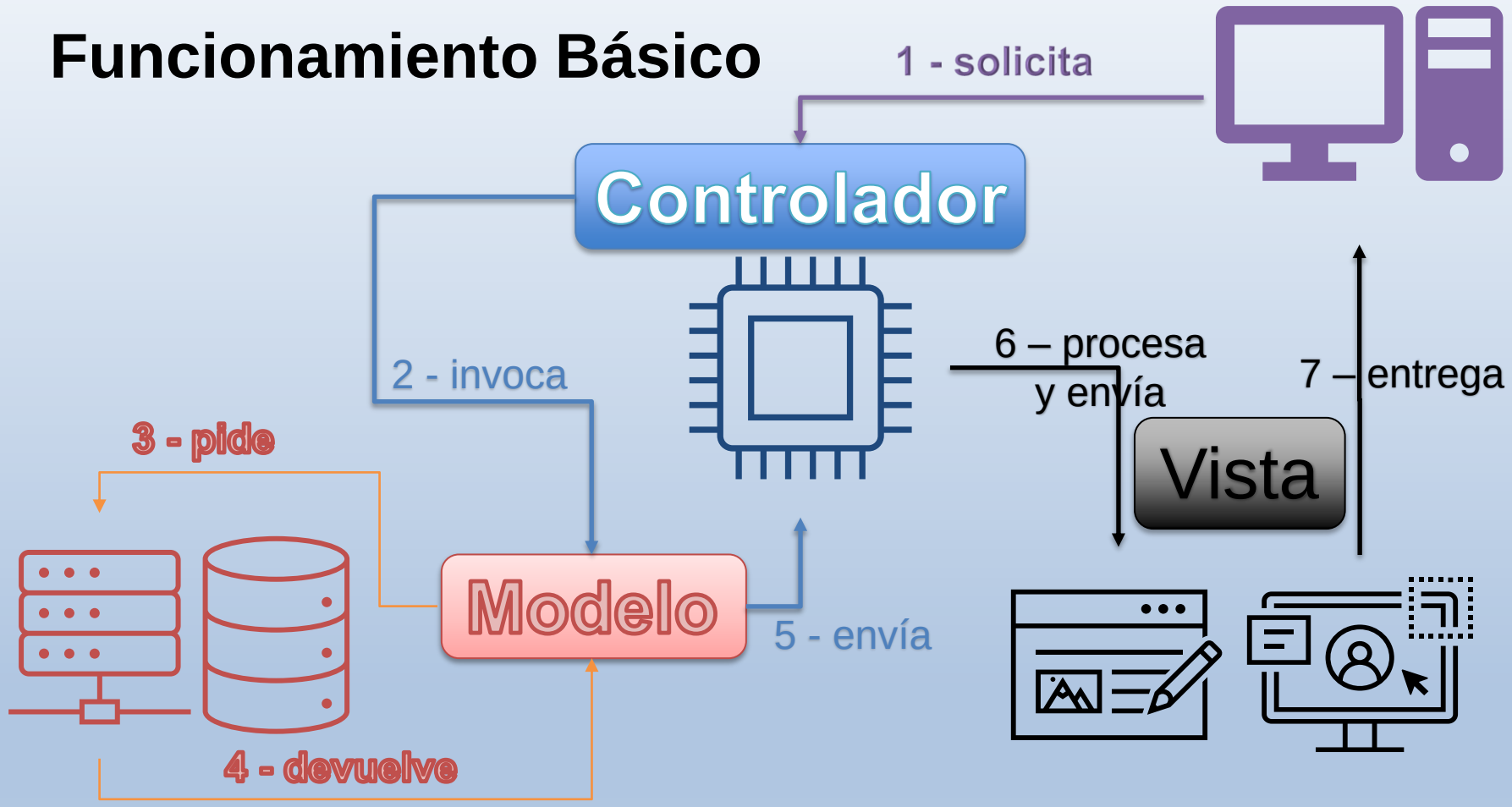
- Vista:
  - Se encarga de mostrar una representación gráfica de los datos al usuario de una forma adecuada.
- Controlador:



  - Es el intermediario entre el modelo y la vista.
  - Controla las peticiones del usuario, solicitando los datos al modelo y mandándoselos a la vista.

MVC

Funcionamiento Básico



Florida Universitària

MVC

Funcionamiento Básico

1. El usuario realiza una petición desde el navegador.
2. El controlador captura el evento.
3. El controlador realiza una llamada al modelo/s correspondientes.
4. El modelo interactúa con la base de datos y retorna la información al controlador.
5. El controlador recibe la información que procesa y la envía a la vista.
6. La vista, procesa la información.
   a) Capa de abstracción para la lógica (procesa datos).
   b) Capa para el diseño de la interfaz gráfica(los “acomoda” al GUI)

MVC

MVC Sencillo

1. Crear un diccionario de datos.

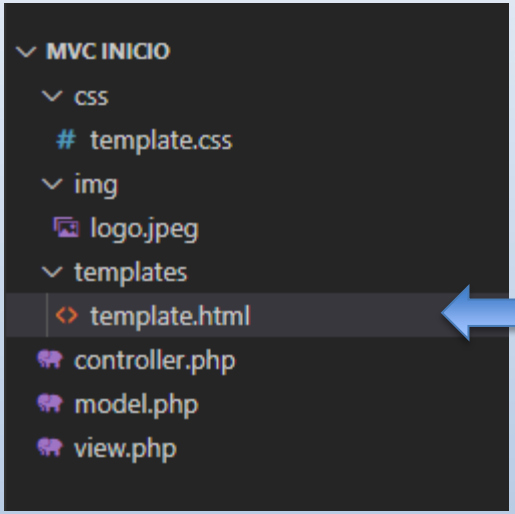
   Qué referencias deben ser reemplazadas por qué datos.
2. Obtener la plantilla HTML
   - Traer la plantilla HTML previamente diseñada, al código PHP.
   - La plantilla debe definir las referencias. Por ejemplo con **{ }**
3. Reemplazar el contenido

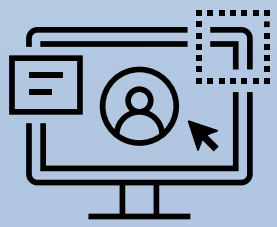
   Hacer la sustitución de referencias por datos en la plantilla.
4. Mostrar el contenido final al usuario.

   2. Salida de la plantilla ***<<renderizada>>***

MVC

MVC Sencillo



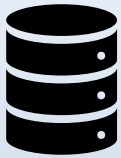


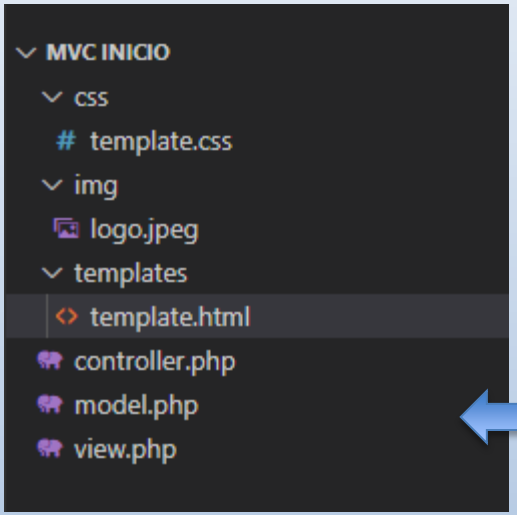
```html
<!DOCTYPE HTML>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;charset=UTF-8">
        <title>{page_title}</title>
        <meta name="title" content="{page_title}">
        <meta name="keywords" content="{keywords}">
        <meta name="description" content="{description}">
        <link type="text/css" rel="stylesheet" href="/css/template.css">
    </head>
    <body>
        <div id="page">
            <div id="cab">
                <img alt="home page" src="/img/logo.jpeg"></a>
            </div>
            <div id="titulo">
                <h1>{page_title}</h1>
            </div>
            <div id="contenido">
                {content}
            </div>
        </div>
    </body>
</html>
```


Florida Universitària

MVC

MVC Sencillo

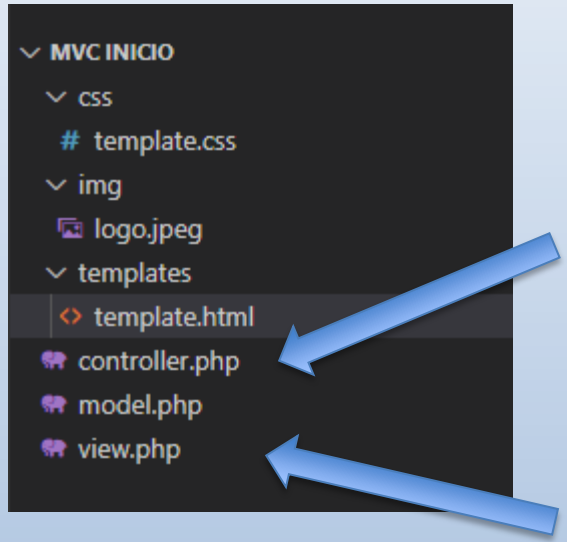


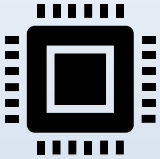



```php
<?php
    // modelo -> proporciona los datos necesarios a la aplicación
    $diccionario = array(
                        'page_title'=>'MVC en PHP',
                        'keywords'=>'poo, mvc, php, arquitectura de software',
                        'description'=>'ponemos en práctica el MVC en PHP',
                        'content'=>'Lorem ipsum dolor sit amet, consectetur
                        adipiscing elit. In ut lorem eu justo blandit laoreet.
                        Curabitur sem nisl, eleifend fermentum sem nec,
                        condimentum sodales mauris. Duis facilisis hendrerit
                        fermentum. Suspendisse venenatis ipsum nisl, nec
                        scelerisque velit faucibus ac. Aliquam erat volutpat.
                        Donec quis enim non risus gravida semper eget eget
                        tellus. Sed fermentum cursus consectetur.'
    );
?>
```

MVC

MVC Sencillo





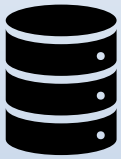
```php
<?php
    require('model.php'); //pide al modelo los datos necesarios
    require('view.php') //envia a la vista los datos para su visualización
?>
```

```php
<?php
    $template = file_get_contents('templates/template.html');
    foreach ($diccionario as $clave=>$valor) {
        $template = str_replace('{'.$clave.'}', $valor, $template);
    };
    echo $template;
?>
```

MVC

MVC Sencillo

1. **Modelo**

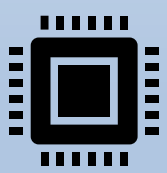


- Debería contener la interacción con la BB.DD.
- Debería implementar métodos como mínimo para:
  - Conectar, desconectar.
  - CRUD (Create, Read, Update, Delete)

2. **Vista**

- Debería poder renderizar diferentes situaciones además de las del tipo de referencia => valor, como estructuras repetitivas, condicionales, etc…

3. **Controlador**



- Debería poder renderizar diferentes plantillas, en función de diferentes peticiones del usuario.
- Necesitamos un **<<Front Controller>>**

MVC

Bibliografía

- MVC Mozilla: https://developer.mozilla.org/es/docs/Glossary/MVC
- Desarrolloweb: https://desarrolloweb.com/articulos/que-es-mvc.html
- Creador: http://heim.ifi.uio.no/~trygver/themes/mvc/mvc-index.html
- Dive Into Desing Patterns – Alexander Shvets.