Tema 4 – Spring MVC

Grado en Ingeniería Informática en Tecnologías de la Información

Departamento de Ingeniería de Sistemas Informáticos y Telemáticos

Área de Lenguajes y Sistemas Informáticos

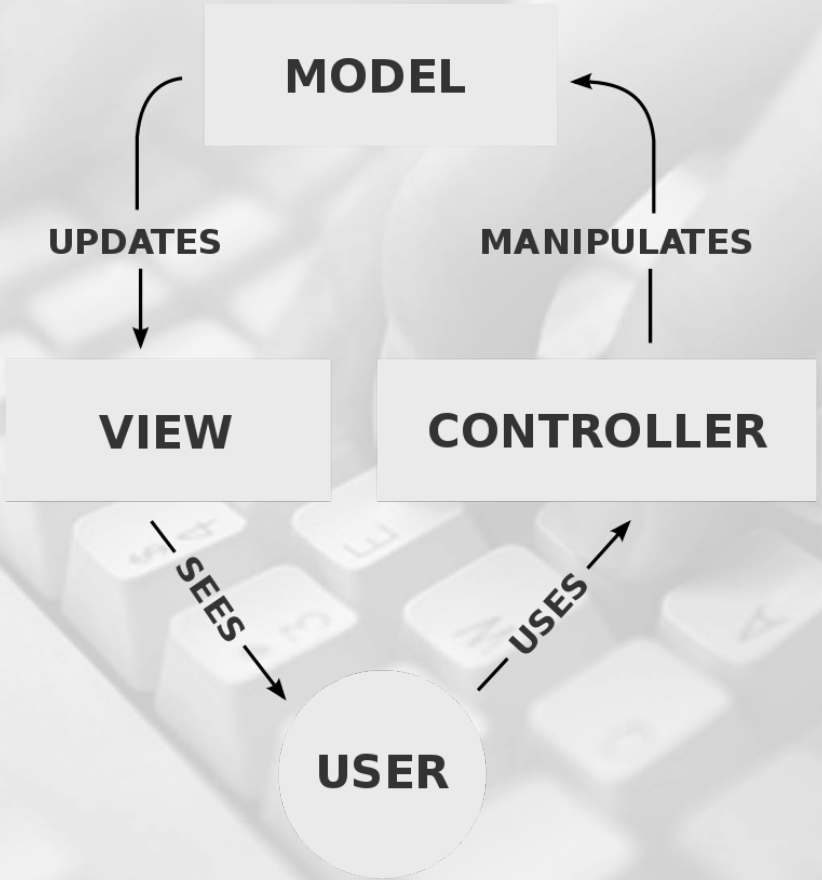
Dr. Luis V. Calderita

Introducción

- El Modelo Vista Controlador:
- El modelo vista controlador (MVC) es un patrón muy popular para construir aplicaciones web basadas en Interfaces de Usuarios (UI-based web-app)
- MVC permite desacoplar el diseño de la aplicación en modelo, vista y controlador

Patrón arquitectónico MVC

- Modelo Vista Controlador.



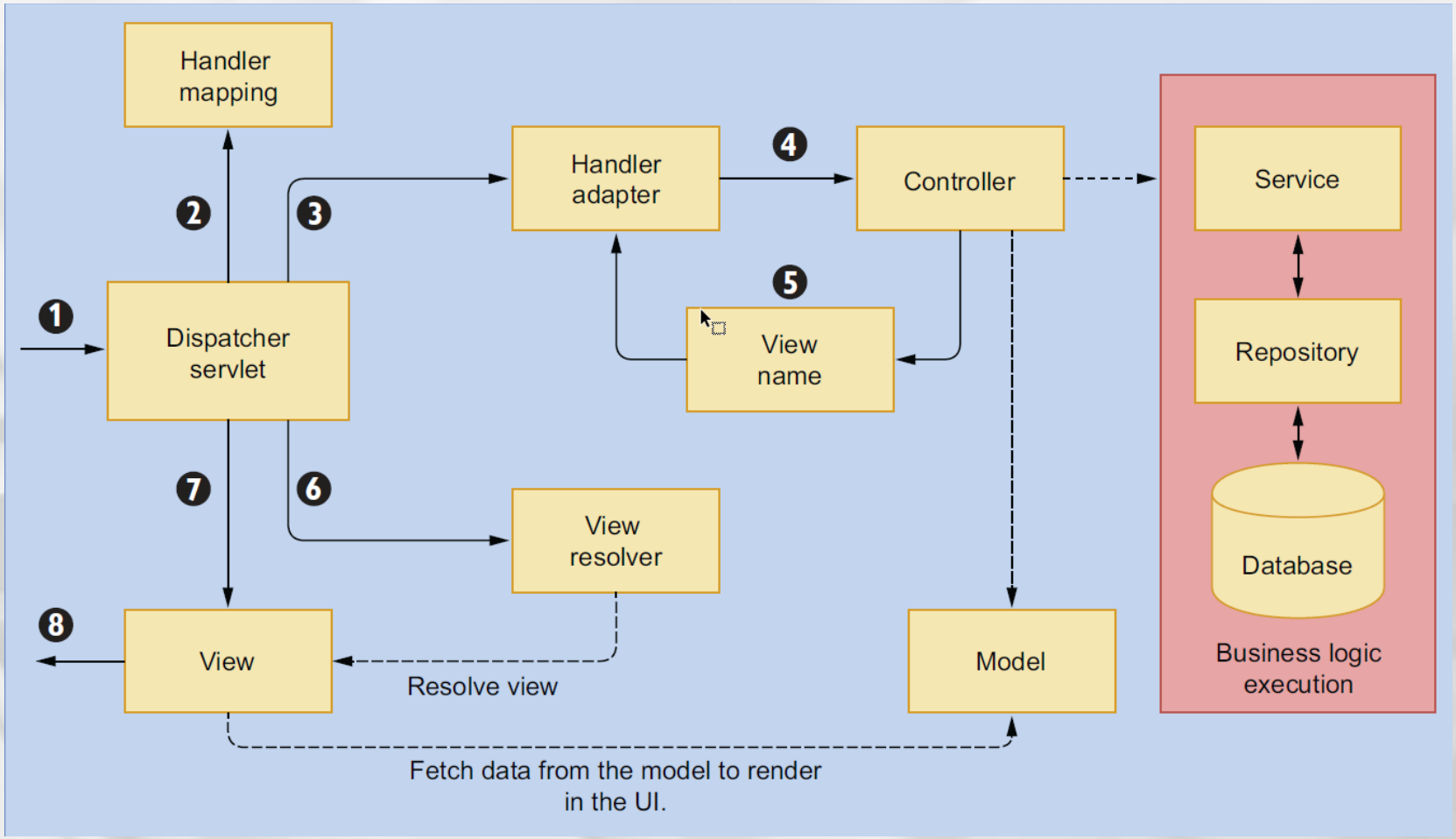
Spring MVC

- Spring MVC es la implementación del patrón de diseño MVC. Es Uno de los módulos más importantes de Spring
- El Modelo (*Model*) **encapsula los datos de la capa de negocio**, la cuál será presentada mediante una Vista (*View*)
- El Controlador (*Controller*) es el responsable de manejar las peticiones del usuario y de invocar a los servicios (al backend)
- Tras la invocación del servicio, el controlador prepara el modelo con los datos para que las vistas se *rendericen* (representen) en la UI.

Spring MVC: Front Controller Design Pattern

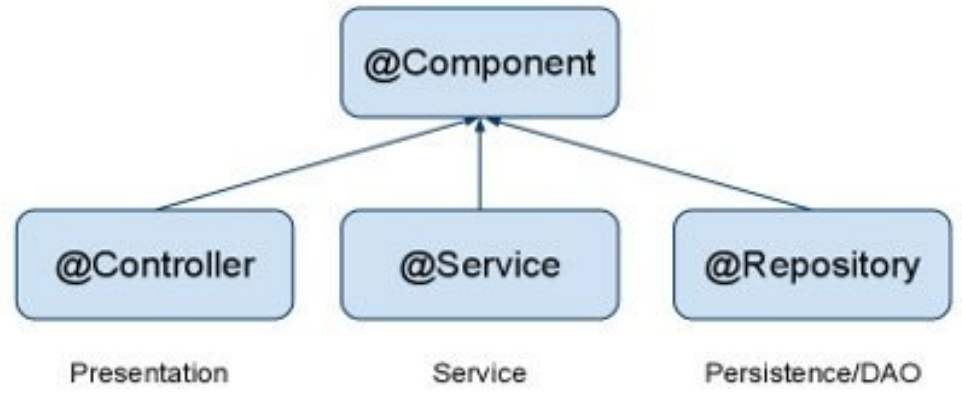
- Spring MVC se basa en el patrón de diseño: *Front Controller Pattern*
- En este patrón de diseño, un servlet central es el principal responsable de manejar todas las peticiones
  - En Spring, este servlet central se conoce como *Dispatcher Servlet*
- Sin embargo, delega la tarea del procesamiento de la petición actual en varios componentes configurables

Componentes de Spring MVC



Estereotipos de Component

- @Service, @Repository y @Controller:
  - Son estereotipos de @Component y se usan para indicar que la clase será un servicio (@Service), una clase de acceso a datos (@Repository) o un controlador (@Controller).



Clase anotada con @Controller

- Funcionalidad General
  - Recibir peticiones http, conectar con la capa de servicios, devolver la vista con la información
- Declara el tipo de método que atenderá a las peticiones de las URL correspondientes
- Devuelve mediante un ***String*** el nombre de la vista

Anotaciones para las peticiones

- Todas las peticiones se pueden mapear con @RequestMapping
- Pero hay anotaciones específicas para los métodos más comunes:
  - @PostMapping - Create
  - @GetMapping - Read
  - @PutMapping – Update (reemplazar)
  - @DeleteMapping - Delete

@Controller, ejemplo básico

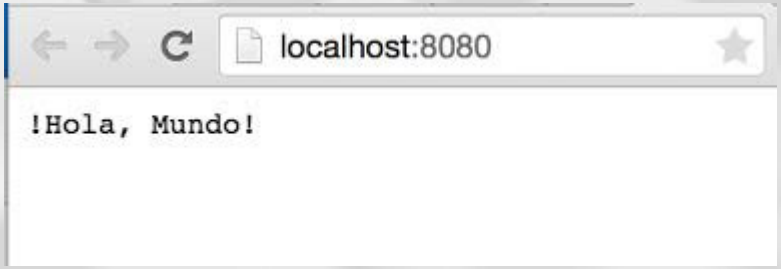
```java
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
@Controller
public class HomeController {

    public HomeController() {
        // TODO Auto-generated constructor stub
        System.out.println("\t HomeController builder");
    }
    @GetMapping("/")
    public String index() {
        System.out.println("\t Recogo la peticion.");
        return "myIndex.html";
    }
}
```

- Petición GET a / (localhost:8080)
- Redirección a una vista estática llamada myIndex.html ubicada en la carpeta static

Practicando

- Añadir un controlador al ejemplo del tema anterior, en el que incluimos un servicio y CommandLineRuner.
  - Este controlador **simplemente** redirigirá localhost:8080 a myIndex.html
  - Crear el fichero myIndex.html con el típico: ¡Hola Mundo!
  - Añadir el fichero myIndex.html a la carpeta **resources/static**
  - Ejecutar la aplicación y comprobar que funciona



Recursos

- Documentación Spring Web MVC
  - Spring Web MVC