Spring MVC

Para usar **Spring MVC** sin **Spring Boot**, necesitas configurar manualmente el framework de Spring, ya que Spring Boot generalmente automatiza gran parte de la configuración. A continuación se presenta una guía paso a paso para crear una simple aplicación web de Spring MVC utilizando una configuración basada en XML. Este ejemplo configurará una aplicación básica con un controlador que muestra un mensaje de "Bienvenido"en una página JSP.

Guía Paso a Paso

1. Configurar la Estructura del Proyecto

Crea una estructura estándar de aplicación web. Puedes usar un IDE como IntelliJ o Eclipse, o crear manualmente las carpetas. La estructura debe verse así:

```
SimpleSpringMVCApp/
  src/
     main/
         java/
             com/
                 example/
                     controllers/
                         HomeController.java
         webapp/
             WEB-INF/
                 web.xml
                 spring-mvc-config.xml
                 views/
                     home.jsp
  pom.xml  (si se usa Maven)
```

- `src/main/java`: Contiene tu código fuente de Java (por ejemplo, controladores).
- `src/main/webapp/WEB-INF`: Contiene archivos de configuración (`web.xml`, `spring-mvc-config.xml`) y vistas JSP.

2. Agregar Dependencias

Si usas Maven, incluye las dependencias necesarias en tu `pom.xml`. Para una simple aplicación Spring MVC, necesitas la biblioteca Spring Web MVC y la API Servlet (proporcionada por el contenedor).

Crea o edita `pom.xml` con el siguiente contenido:

```xml
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.example</groupId>
    <artifactId>SimpleSpringMVCApp</artifactId>
    <version>1.0-SNAPSHOT</version>
    <packaging>war</packaging>

    <dependencies>
        <!-- Spring Web MVC -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-webmvc</artifactId>
            <version>5.3.10</version>
        </dependency>
        <!-- Servlet API (proporcionada por el contenedor) -->
        <dependency>
            <groupId>javax.servlet</groupId>
            <artifactId>javax.servlet-api</artifactId>
            <version>4.0.1</version>
            <scope>provided</scope>
        </dependency>
    </dependencies>

    <build>
        <plugins>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-war-plugin</artifactId>
                <version>3.3.1</version>
            </plugin>
        </plugins>
    </build>
</project>
```

- **Notas**:
  - `<packaging>war</packaging>`: Asegura que el proyecto se empaquete como un archivo WAR para su implementación en un contenedor de servlets.

- Si no usas Maven, descarga manualmente los archivos JAR de Spring MVC y Servlet API y agrégalos al classpath de tu proyecto.

3. Configurar el DispatcherServlet en `web.xml`

El archivo `web.xml` es el descriptor de implementación de tu aplicación web. Configura el `DispatcherServlet`, el controlador frontal de Spring MVC, para manejar las solicitudes entrantes.

Crea `src/main/webapp/WEB-INF/web.xml` con el siguiente contenido:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0
         version="4.0">

    <!-- Define el DispatcherServlet -->
    <servlet>
        <servlet-name>spring-mvc</servlet-name>
        <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
        <init-param>
            <param-name>contextConfigLocation</param-name>
            <param-value>/WEB-INF/spring-mvc-config.xml</param-value>
        </init-param>
        <load-on-startup>1</load-on-startup>
    </servlet>

    <!-- Mapa el DispatcherServlet para manejar todas las solicitudes -->
    <servlet-mapping>
        <servlet-name>spring-mvc</servlet-name>
        <url-pattern>/</url-pattern>
    </servlet-mapping>
</web-app>
```

- **Explicación**:
  - `<servlet-class>`: Especifica el `DispatcherServlet`, que enruta las solicitudes a los controladores.
  - `<init-param>`: Apunta al archivo de configuración de Spring (`spring-mvc-config.xml`).
  - `<url-pattern>/</url-pattern>`: Mapa el servlet para manejar todas las solicitudes a la aplicación.

4. Crear el Archivo de Configuración de Spring

Crea `src/main/webapp/WEB-INF/spring-mvc-config.xml` para definir los beans de Spring MVC, como controladores y resolutores de vistas.

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:context="http://www.springframework.org/schema/context"
       xmlns:mvc="http://www.springframework.org/schema/mvc"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="
           http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-bean
           http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-
           http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc.xsd">

    <!-- Habilita el escaneo de componentes para controladores -->
    <context:component-scan base-package="com.example.controllers" />

    <!-- Habilita la configuración MVC basada en anotaciones -->
    <mvc:annotation-driven />

    <!-- Configura el resolutor de vistas para archivos JSP -->
    <bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <property name="prefix" value="/WEB-INF/views/" />
        <property name="suffix" value=".jsp" />
    </bean>
</beans>
```

- **Explicación**:
  - `<context:component-scan>`: Escanea el paquete `com.example.controllers` para componentes anotados (por ejemplo, `@Controller`).
  - `<mvc:annotation-driven>`: Habilita las características MVC basadas en anotaciones (por ejemplo, `@GetMapping`).
  - `InternalResourceViewResolver`: Mapa los nombres de vistas a archivos JSP en `/WEB-INF/views/` con un sufijo `.jsp`.

5. Crear un Controlador Simple

Crea un controlador para manejar las solicitudes HTTP. Agrega `HomeController.java` en `src/main/java/com/example/con`

```java
package com.example.controllers;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;

@Controller
public class HomeController {

    @GetMapping("/")
    public String home() {
        return "home";
    }
}
```

- **Explicación**:
  - `@Controller`: Marca esta clase como un controlador de Spring MVC.
  - `@GetMapping("/")`: Mapa las solicitudes GET a la URL raíz (`/`) al método `home()`.
  - `return "home"`: Devuelve el nombre de la vista `"home"`, que se resuelve a `/WEB-INF/views/home.jsp`.

6. Crear una Vista JSP

Crea un archivo JSP simple para mostrar la salida. Agrega `home.jsp` en `src/main/webapp/WEB-INF/views/`:

```jsp
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
<head>
    <title>Home</title>
</head>
<body>
    <h1>Bienvenido a Spring MVC sin Spring Boot</h1>
</body>
</html>
```

7. Construir y Empaquetar la Aplicación

Si usas Maven, ejecuta el siguiente comando desde la raíz del proyecto para construir el archivo WAR:

```
mvn clean package
```

Esto genera `SimpleSpringMVCApp-1.0-SNAPSHOT.war` en el directorio `target`.

- **Nota**: Si no usas Maven, compila los archivos Java y empaqueta el proyecto en un archivo WAR manualmente, manteniendo la estructura de carpetas.

8. Implementar la Aplicación

Implementa el archivo WAR en un contenedor de servlets como Apache Tomcat: - Copia `SimpleSpringMVCApp-1.0-SNAPS` al directorio `webapps` de tu instalación de Tomcat. - Inicia Tomcat (por ejemplo, ejecuta `startup.sh` o `startup.bat` desde el directorio `bin` de Tomcat).

9. Acceder a la Aplicación

Abre un navegador y navega a:

```
http://localhost:8080/SimpleSpringMVCApp-1.0-SNAPSHOT/
```

Deberías ver:

```
Bienvenido a Spring MVC sin Spring Boot
```

- **Nota**: La ruta del contexto (`/SimpleSpringMVCApp-1.0-SNAPSHOT`) depende del nombre del archivo WAR. Si renombras el WAR a `myapp.war`, la URL se convierte en `http://localhost:8080/myapp/`.

Resumen

Para usar **Spring MVC sin Spring Boot**: 1. **Configura un proyecto web** con una estructura estándar. 2. **Agrega dependencias** para Spring MVC y la API Servlet. 3. **Configura el DispatcherServlet** en `web.xml`. 4. **Define la configuración de Spring** en un archivo XML (por ejemplo, `spring-mvc-config.xml`). 5. **Crea un controlador** para manejar las solicitudes. 6. **Agrega una vista JSP** para la respuesta. 7. **Construye y despliega** el archivo WAR en un contenedor de servlets.

Este enfoque te da el control total sobre la configuración, ¡haciéndolo una excelente manera de aprender los fundamentos de Spring MVC!