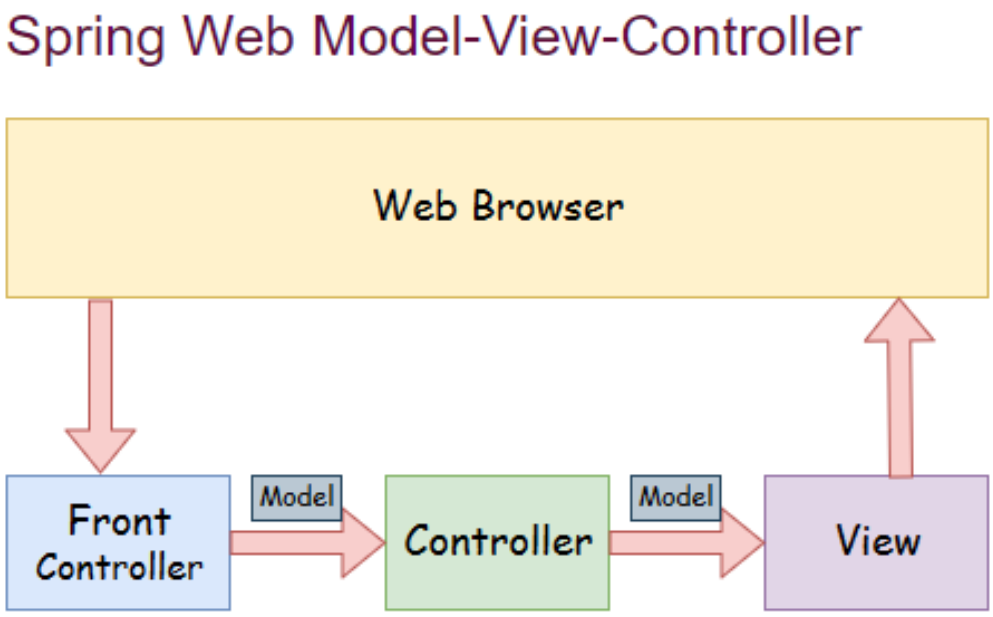
Spring MVC

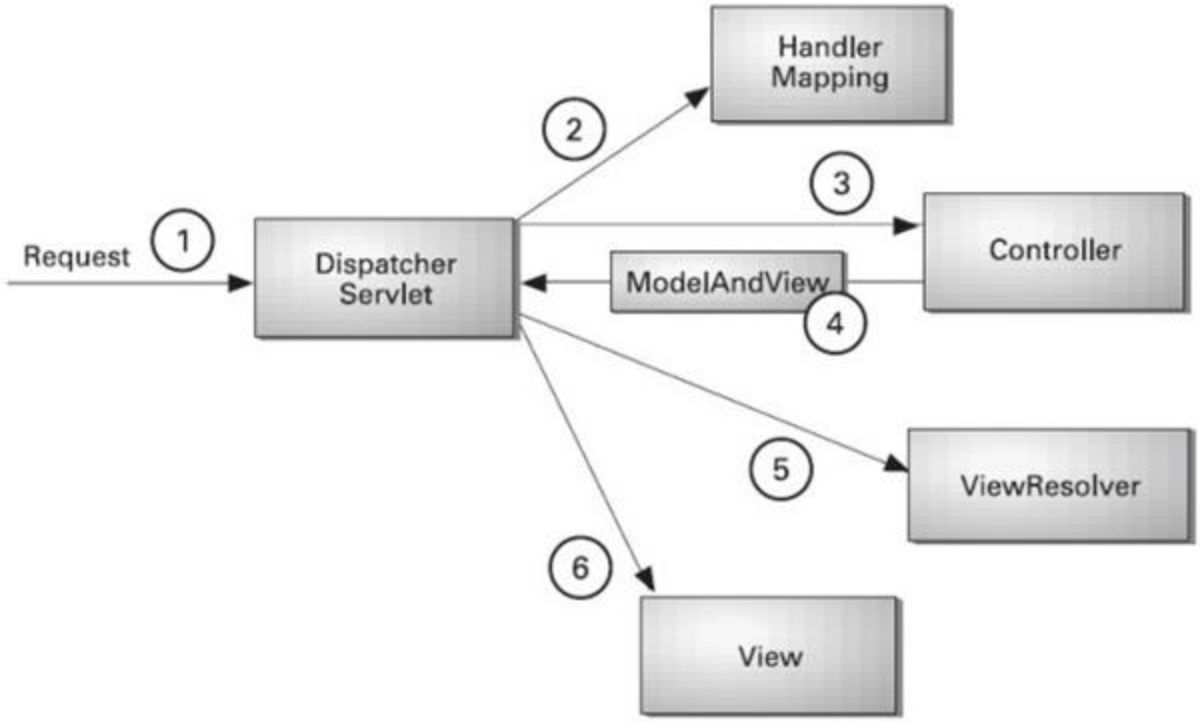
A Spring MVC is a Java framework which is used to build web applications. It follows the Model-View-Controller design pattern. It implements all the basic features of a core spring framework like Inversion of Control, Dependency Injection.

A Spring MVC provides an elegant solution to use MVC in spring framework by the help of **DispatcherServlet**. Here, **DispatcherServlet** is a class that receives the incoming request and maps it to the right resource such as controllers, models, and views.



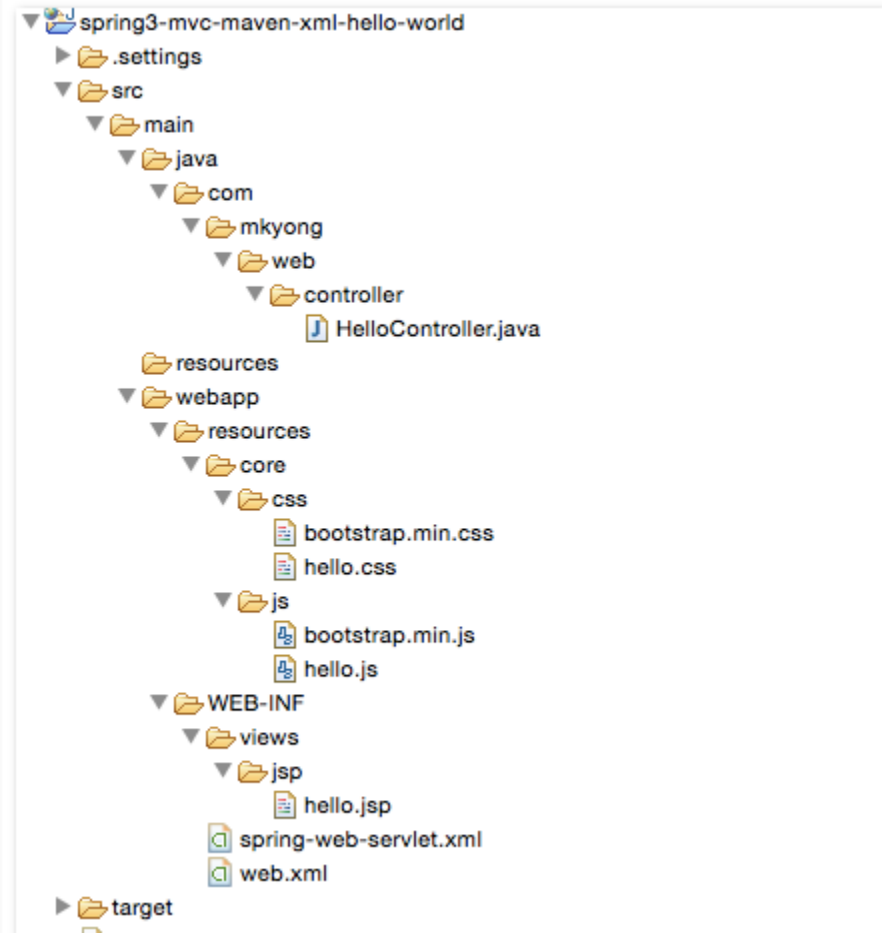
- **Model** - A model contains the data of the application. A data can be a single object or a collection of objects.
- **Controller** - A controller contains the business logic of an application. Here, the @Controller annotation is used to mark the class as the controller.
- **View** - A view represents the provided information in a particular format. Generally, JSP is used to create a view page. Although spring also supports other view technologies such as Apache Velocity, Thymeleaf and FreeMarker.
- **Front Controller** - In Spring Web MVC, the DispatcherServlet class works as the front controller. It is responsible to manage the flow of the Spring MVC application.

Understanding the flow of Spring Web MVC



- As displayed in the figure, all the incoming request is intercepted by the DispatcherServlet that works as the front controller.
- The DispatcherServlet gets an entry of handler mapping from the XML file and forwards the request to the controller.
- The controller returns an object of ModelAndView.
- The DispatcherServlet checks the entry of view resolver in the XML file and invokes the specified view component.

We are building hello world application using Spring MVC framework. Follow the given instructions step by step and learn the basics.



pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<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.javaTraining</groupId>
  <artifactId>springmvc</artifactId>
  <version>1.0-SNAPSHOT</version>
  <packaging>war</packaging>

  <properties>
    <spring.version>4.0.5.RELEASE</spring.version>
  </properties>
```

```
  <dependencies>
    <dependency>
      <groupId>org.springframework</groupId>
      <artifactId>spring-core</artifactId>
      <version>${spring.version}</version>
    </dependency>
    <dependency>
      <groupId>org.springframework</groupId>
      <artifactId>spring-web</artifactId>
      <version>${spring.version}</version>
    </dependency>
    <dependency>
      <groupId>org.springframework</groupId>
      <artifactId>spring-webmvc</artifactId>
      <version>${spring.version}</version>
    </dependency>
  </dependencies>

  <!--<build>
    <plugins>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-compiler-plugin</artifactId>
        <configuration>
          <source>1.8</source>
          <target>1.8</target>
        </configuration>
      </plugin>
    </plugins>
  </build>
-->
</project>
```

Web.xml

```
<web-app id = "WebApp_ID" version = "2.4"
      xmlns = "http://java.sun.com/xml/ns/j2ee"
      xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation = "http://java.sun.com/xml/ns/j2ee
        http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">

  <display-name>Spring MVC Application</display-name>

  <servlet>
    <servlet-name>HelloWeb</servlet-name>
```

```
      <servlet-class>
         org.springframework.web.servlet.DispatcherServlet
      </servlet-class>
      <load-on-startup>1</load-on-startup>
   </servlet>

   <servlet-mapping>
      <servlet-name>HelloWeb</servlet-name>
      <url-pattern>/</url-pattern>
   </servlet-mapping>


</web-app>
```

Bean configuration file

We will need to write configuration file (`spring-mvc-servlet.xml`) inside `WEB-INF` folder. We can name it at our will, but keep remember that it must match the servlet name we declared in `web.xml`.

HelloWeb-servlet.xml

```
<beans xmlns = "http://www.springframework.org/schema/beans"
           xmlns:context = "http://www.springframework.org/schema/context"
           xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
           xsi:schemaLocation = "http://www.springframework.org/schema/beans
  http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
  http://www.springframework.org/schema/context
  http://www.springframework.org/schema/context/spring-context-3.0.xsd">

<context:component-scan base-package = "com.javaTraining" />

<bean class = "org.springframework.web.servlet.view.InternalResourceViewResolver">
   <property name = "prefix" value = "/WEB-INF/jsp/" />
   <property name = "suffix" value = ".jsp" />
</bean>

</beans>
```

JSP Views

```
<%@ page contentType = "text/html; charset = UTF-8" %>
<html>
  <head>
    <title>Hello World</title>
  </head>

  <body>
    <h2>${message}</h2>
  </body>
</html>
```

HelloController.java

```
package com.javaTraining;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.ui.ModelMap;

@Controller
@RequestMapping("/hello")
public class HelloController {
   @RequestMapping(method = RequestMethod.GET)public String printHello(ModelMap model) {
      model.addAttribute("message", "Hello Spring MVC Framework!");
      return "hello";
   }
}
```