Spring MVC : introduction

Achref El Mouelhi

Docteur de l'université d'Aix-Marseille
Chercheur en programmation par contrainte (IA)
Ingénieur en génie logiciel

elmouelhi.achref@gmail.com

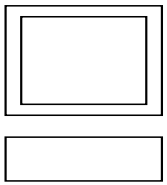
Plan

Spring MVC 5

Spring MVC

- un des frameworks **Spring**
- permettant de simplifier la création d'applications web
- masquant l'utilisation de l'**API Servlet** de **JEE** grâce aux annotations `@RequestParam`...
- implémentant le patron de conception **MVC 2**
- basé sur le concept d'injection de dépendances
- facilitant le développement de micro-services **REST**

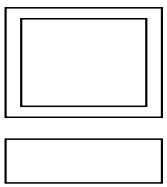
Spring MVC 5



Client

Serveur

Spring MVC 5

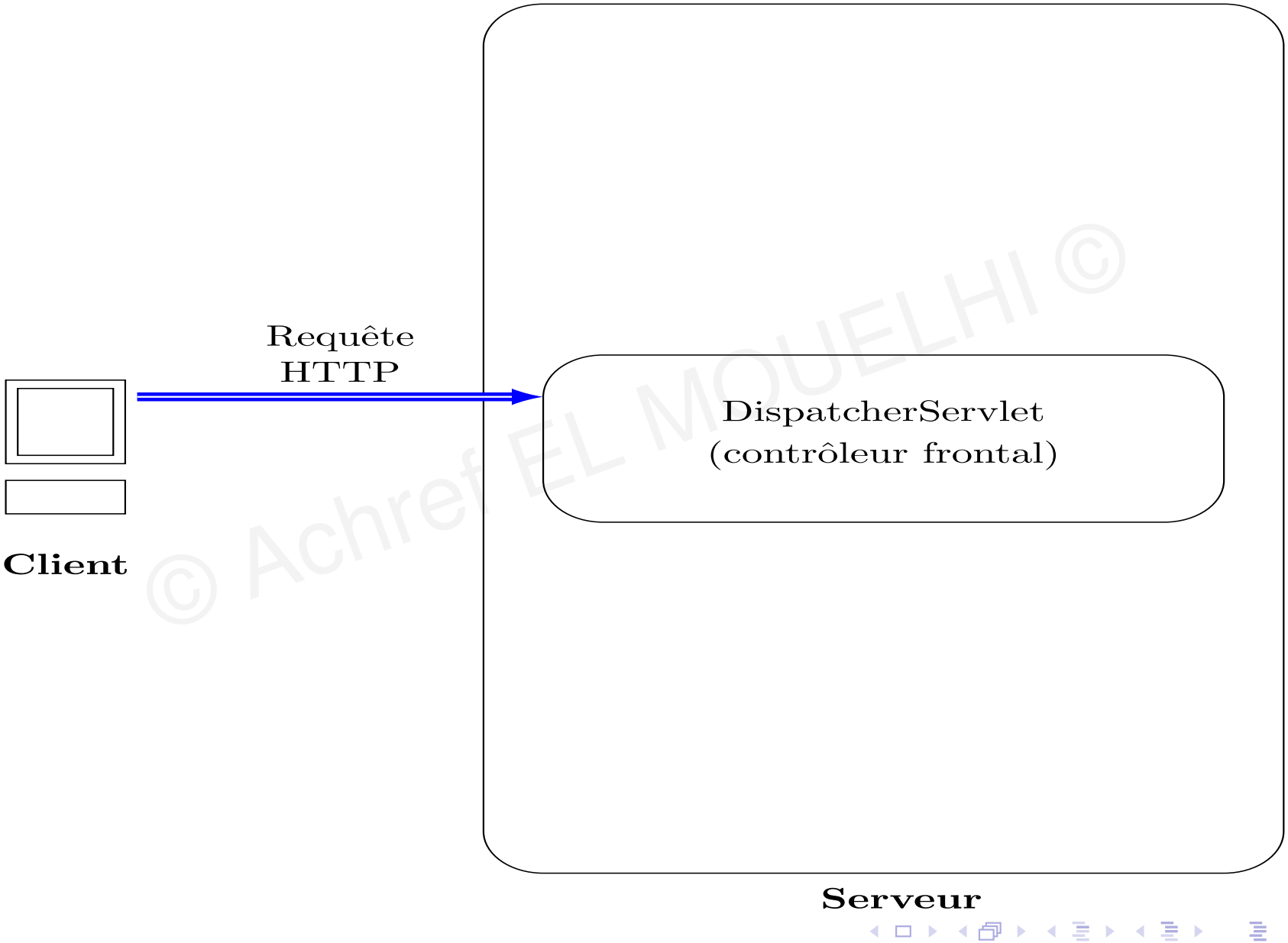


Client

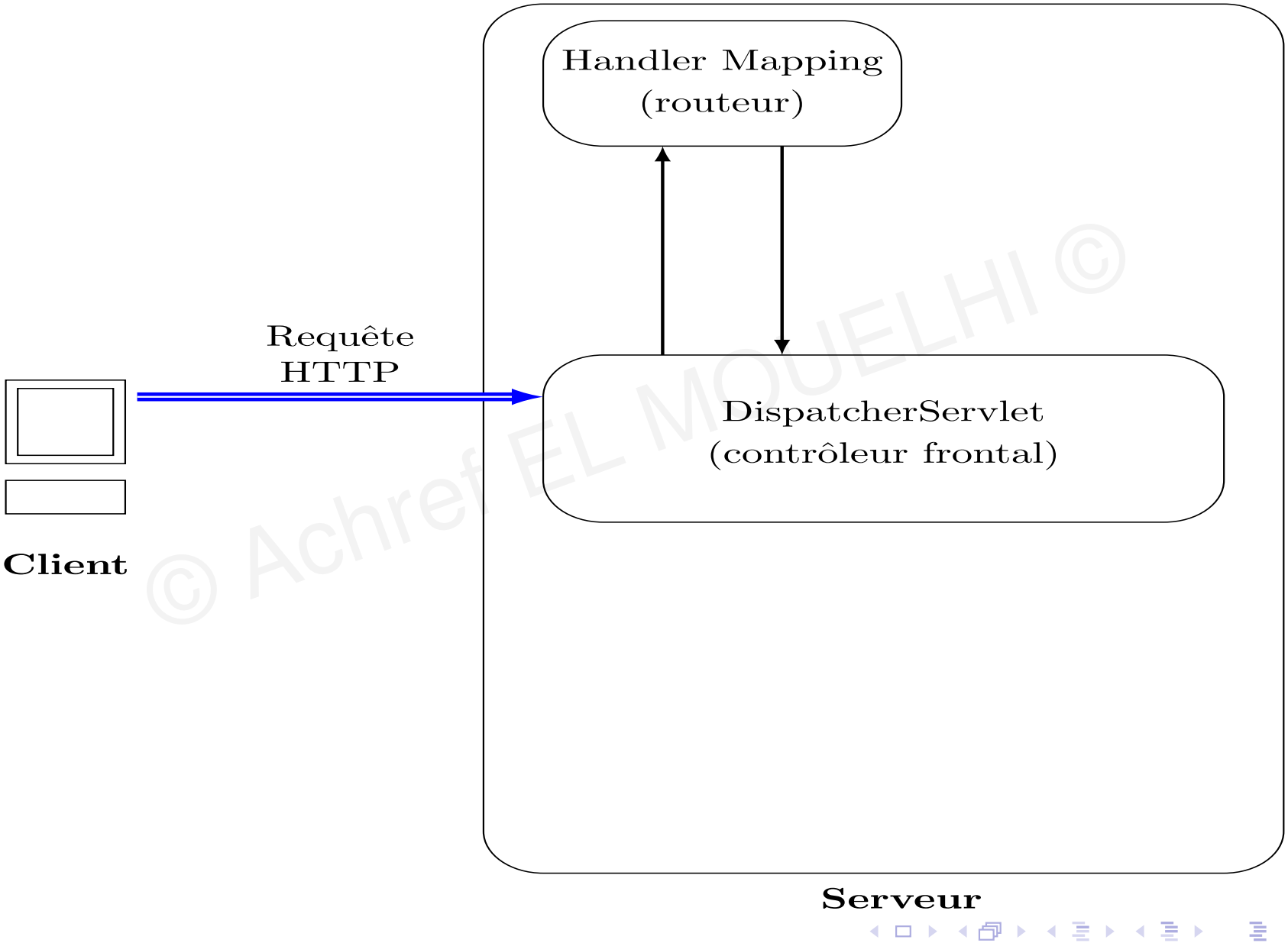
DispatcherServlet
(contrôleur frontal)

Serveur

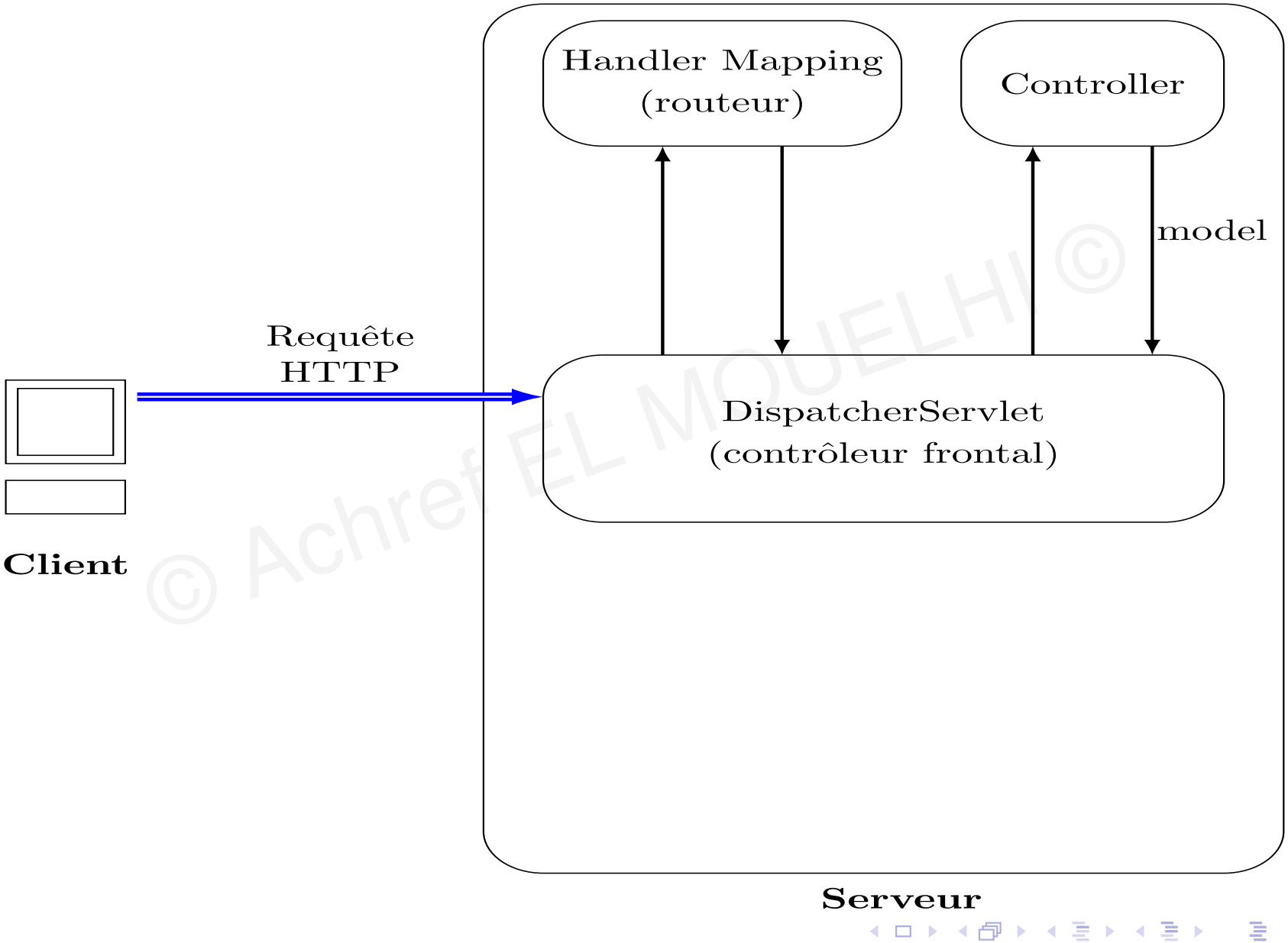
Spring MVC 5



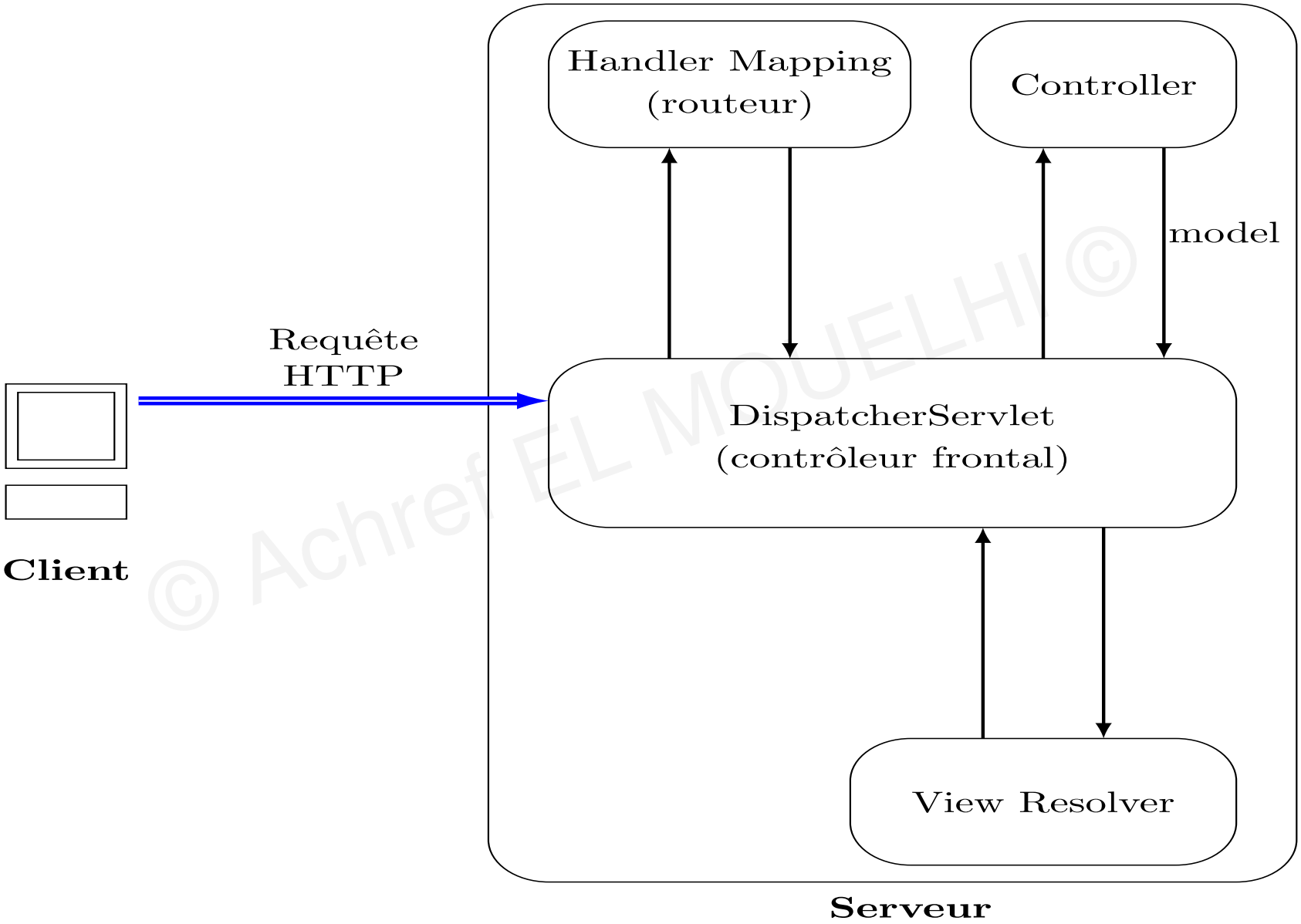
Spring MVC 5



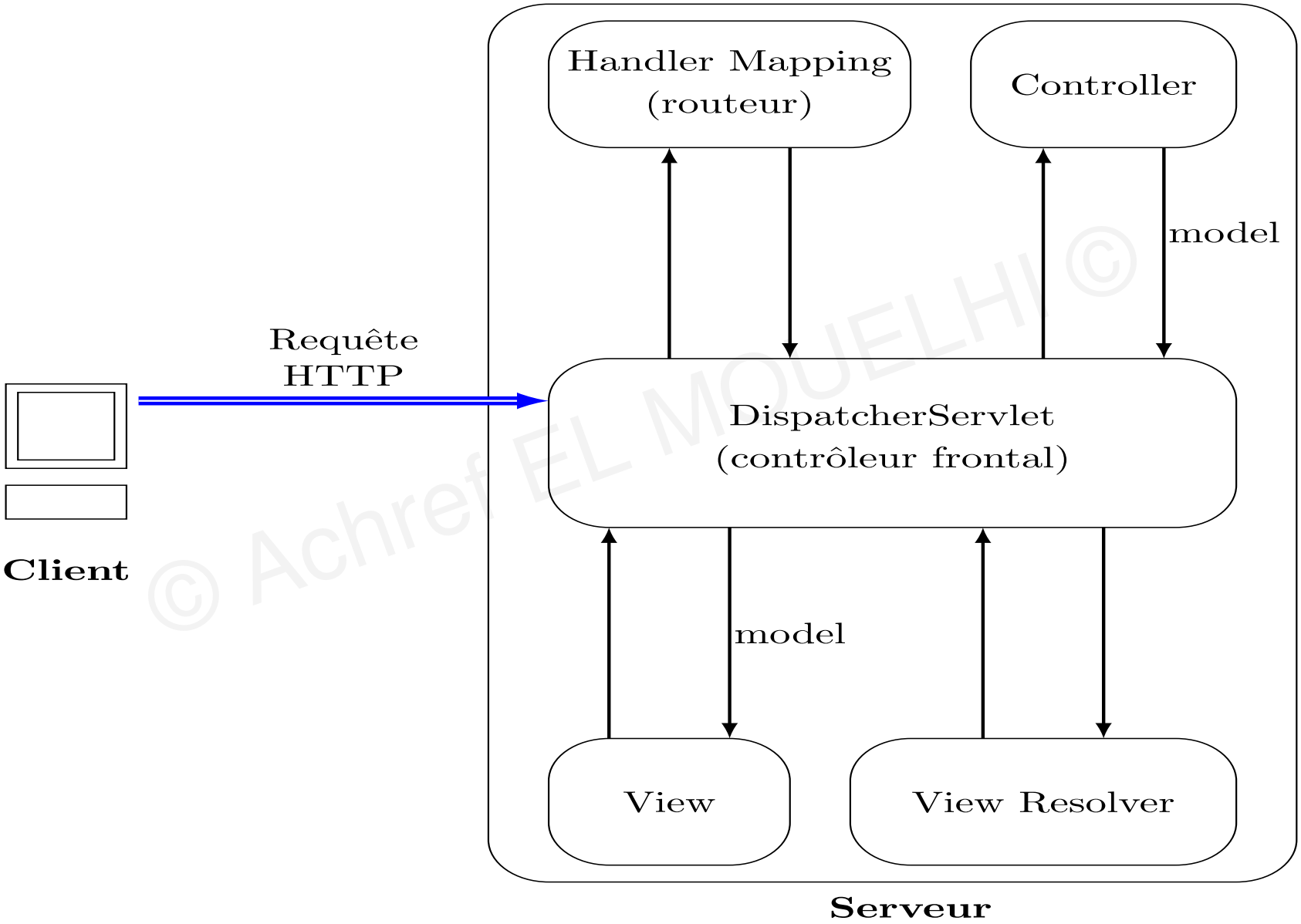
Spring MVC 5



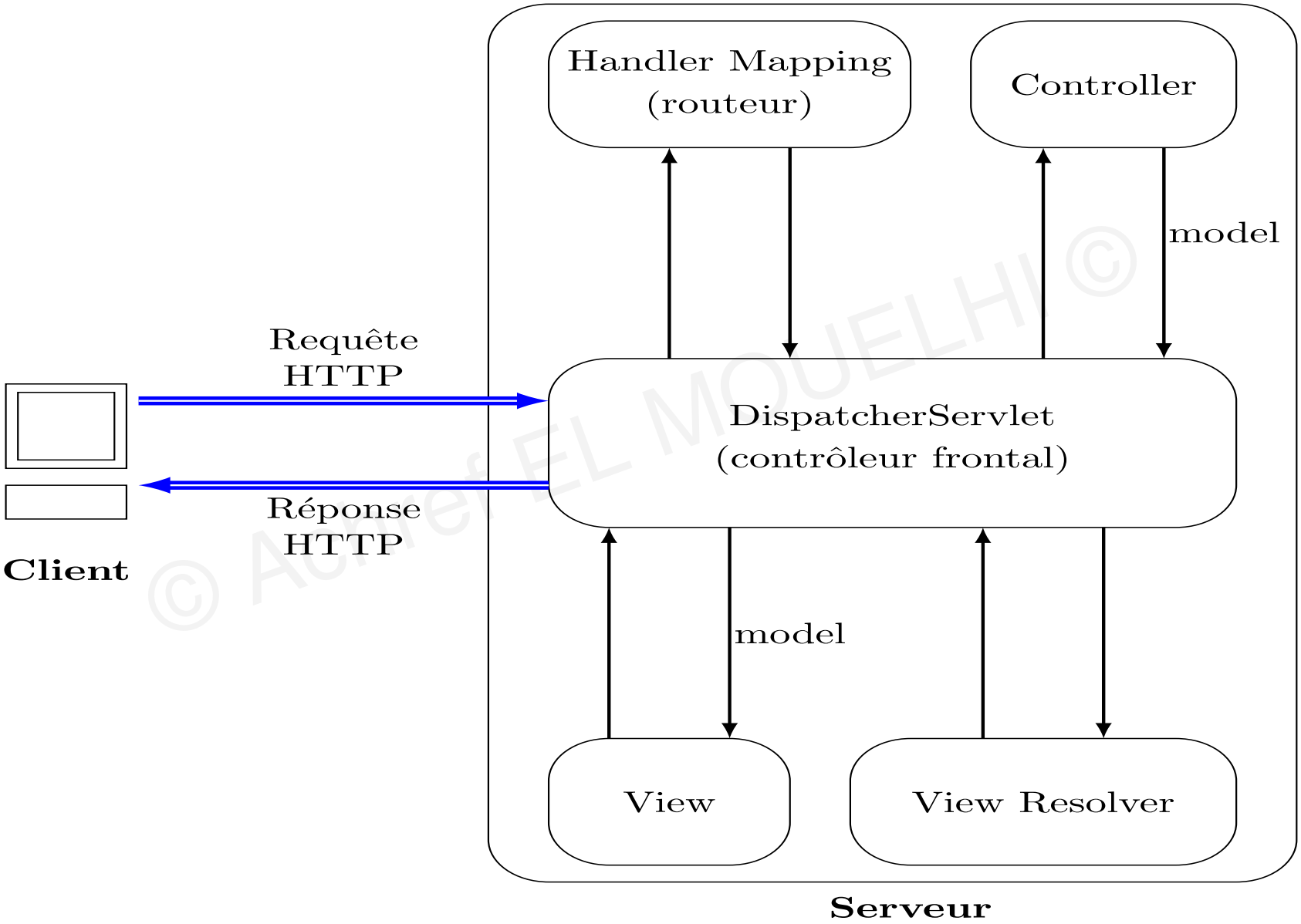
Spring MVC 5



Spring MVC 5



Spring MVC 5



Spring MVC 5

Explication

- Le contrôleur frontal intercepte la requête **HTTP** du client, communique avec le `Handler mapping`, pour la rediriger vers le contrôleur approprié (selon la route).
- Le contrôleur traite la requête, prépare un modèle et retourne le nom d'une vue au contrôleur frontal.
- Le contrôleur frontal consulte le `View resolver` pour connaître certains détails sur la vue puis transmet le modèle à cette dernière afin de préparer la réponse .
- Le contrôleur frontal retourne une réponse **HTTP** contenant la vue au client.

Spring MVC 5

Créons un projet **Spring MVC** avec dépendances gérées par **Maven**

- Aller dans `File > New > Other`
- Chercher `Spring`
- Choisir `Spring Legacy Project` et cliquer sur `Next`
- Saisir `first-spring-mvc` dans `Project name` puis sélectionner `Spring MVC Project` et cliquer sur `Next`
- Saisir `org.eclipse.firstspringmvc` et valider

Spring MVC 5

Projet marqué par une croix rouge : solution

- Allez dans les propriétés du projet (clic droit puis `properties`) et vérifiez dans `Targeted Runtimes` qu'un serveur est associé au projet
- Faites un clic droit sur le projet et allez dans `Maven > Update Project...`
- Attendez la fin de la mise à jour

Spring MVC 5

Projet marqué par une croix rouge : solution

- Allez dans les propriétés du projet (clic droit puis `properties`) et vérifiez dans `Targeted Runtimes` qu'un serveur est associé au projet
- Faites un clic droit sur le projet et allez dans `Maven > Update Project...`
- Attendez la fin de la mise à jour

Exécuter le projet

`Hello World!` est affiché en allant à `localhost:8080/firstspringmvc/`

Spring MVC 5

Le fichier `web.xml`

- Une application **Spring MVC** utilise les mécanismes de la spécification **Java EE**
- La configuration d'un projet **Java EE**, et aussi **Spring MVC**, passe par un fichier `web.xml` situé dans `WEB-INF`
- Dans ce fichier, et au sein d'une application **Spring MVC**, on trouve la déclaration du contrôleur frontal (`DispatcherServlet`)

Contenu du `web.xml`

```
<web-app ...>
  <context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>/WEB-INF/spring/root-context.xml</param-value>
  </context-param>
  <listener><listener-class>
    org.springframework.web.context.ContextLoaderListener
  </listener-class></listener>
  <servlet>
    <servlet-name>appServlet</servlet-name>
    <servlet-class>org.springframework.web.servlet.DispatcherServlet</
       servlet-class>
    <init-param>
      <param-name>contextConfigLocation</param-name>
      <param-value>/WEB-INF/spring/appServlet/servlet-context.xml</
         param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
  </servlet>
  <servlet-mapping>
    <servlet-name>appServlet</servlet-name>
    <url-pattern>/</url-pattern>
  </servlet-mapping>
</web-app>
```

Spring MVC 5

Explication

- Les attributs de la balise `web-app` sont les namespaces à utiliser dans le projet.
- Dans `<context-param>`, on indique les éléments Spring qui sont partagés par tous les contrôleurs.
- Dans `<listener>`, on charge l'écouteur de Spring.
- Ensuite, on déclare le contrôleur frontal `DispatcherServlet` et on lui redirige toutes les requêtes (avec `/`).
- Le contrôleur frontal charge les paramètres de l'application à partir d'un fichier indiqué dans la balise `<param-value>` de `<init-param>`

Spring MVC 5

Allons voir `servlet-context.xml` (première partie)

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

Explication

- **Spring** commence par charger et affecter le contenu de certaines librairies indispensables pour la validation de ce fichier de configuration
- Ces librairies seront chargées dans les trois `namespace` suivant : `mvc`, `beans` et `context`
- Ces trois `namespace` permettront de configurer une application **Spring MVC**

Spring MVC 5

`servlet-context.xml` **(deuxième partie)**

```
  <!-- Enables the Spring MVC @Controller programming model -->
  <annotation-driven />

  <!-- Handles HTTP GET requests for /resources/** by efficiently
     serving up static resources in the ${webappRoot}/resources
     directory -->
  <resources mapping="/resources/**" location="/resources/" />

  <!-- Resolves views selected for rendering by @Controllers to .jsp
     resources in the /WEB-INF/views directory -->
  <beans:bean class="org.springframework.web.servlet.view.
     InternalResourceViewResolver">
    <beans:property name="prefix" value="/WEB-INF/views/" />
    <beans:property name="suffix" value=".jsp" />
  </beans:bean>

  <context:component-scan base-package="org.eclipse.firstspringmvc" />
</beans:beans>
```

Spring MVC 5

Explication

- La balise `<annotation-driven />` active les annotations permettant d'utiliser un contrôleur Spring MVC telles que `@Controller`...
- La balise `<resources mapping="/resources/**" location="/resources/" />` indique l'emplacement de fichiers statiques (`js`, `css`...). Ce répertoire `resources` est situé dans `webapp`
- Le bean permet d'indiquer que nos vues seront dans le répertoire `views` (situé dans `WEB-INF`) et qui auront comme extension `.jsp`
- Le dernier permet de préciser l'emplacement des contrôleurs de l'application : le package `org.eclipse.firstspringmvc`

Spring MVC 5

Mises à jour à faire

- Restructurer les packages
- Changer la version de **Spring** (5.2.8)
- Changer la version de la **JDK** (13)
- Changer la version de **Dynamic Web Module** (4.0)

Spring MVC 5

Restructuration de packages

- Pour rester cohérent, renommons le package contenant les contrôleurs `org.eclipse.firstspringmvc.controller`
- Pour cela, clic droit sur le package et aller dans `Refactor > Rename`
- Saisir `org.eclipse.firstspringmvc.controller` puis valider

Spring MVC 5

Restructuration de packages

- Pour rester cohérent, renommons le package contenant les contrôleurs `org.eclipse.firstspringmvc.controller`
- Pour cela, clic droit sur le package et aller dans `Refactor > Rename`
- Saisir `org.eclipse.firstspringmvc.controller` puis valider

Vérifions les modifications dans `servlet-context.xml`

```
<context:component-scan
base-package="org.eclipse.firstspringmvc.controller"/>
```

Spring MVC 5

Restructuration de packages

- Pour rester cohérent, renommons le package contenant les contrôleurs `org.eclipse.firstspringmvc.controller`
- Pour cela, clic droit sur le package et aller dans `Refactor > Rename`
- Saisir `org.eclipse.firstspringmvc.controller` puis valider

Vérifions les modifications dans `servlet-context.xml`

```
<context:component-scan
base-package="org.eclipse.firstspringmvc.controller"/>
```

Relancer le projet

Spring MVC 5

Dans `pom.xml`, allons dans la section `properties` (contenu actuel)

```
<properties>
    <java-version>1.6</java-version>
    <org.springframework-version>3.1.1.RELEASE</org.springframework-version>
    <org.aspectj-version>1.6.10</org.aspectj-version>
    <org.slf4j-version>1.6.6</org.slf4j-version>
</properties>
```

Spring MVC 5

Dans `pom.xml`, allons dans la section `properties` (contenu actuel)

```
<properties>
    <java-version>1.6</java-version>
    <org.springframework-version>3.1.1.RELEASE</org.springframework-version>
    <org.aspectj-version>1.6.10</org.aspectj-version>
    <org.slf4j-version>1.6.6</org.slf4j-version>
</properties>
```

Et le remplacer par

```
<properties>
    <java-version>13</java-version>
    <org.springframework-version>5.2.8.RELEASE</org.springframework-version>
    <org.aspectj-version>1.9.6</org.aspectj-version>
    <org.slf4j-version>1.6.6</org.slf4j-version>
</properties>
```

Supprimer la dépendance suivante

```
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-context</artifactId>
  <version>${org.springframework-version}</version>
  <exclusions>
    <!-- Exclude Commons Logging in favor of SLF4j -->
    <exclusion>
      <groupId>commons-logging</groupId>
      <artifactId>commons-logging</artifactId>
    </exclusion>
  </exclusions>
</dependency>
```

La dépendance `spring-webmvc 5` est suffisante pour télécharger `spring-context`.

Supprimer la dépendance suivante

```
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-context</artifactId>
  <version>${org.springframework-version}</version>
  <exclusions>
    <!-- Exclude Commons Logging in favor of SLF4j -->
    <exclusion>
      <groupId>commons-logging</groupId>
      <artifactId>commons-logging</artifactId>
    </exclusion>
  </exclusions>
</dependency>
```

La dépendance `spring-webmvc 5` est suffisante pour télécharger `spring-context`.

Relancer le projet

Spring MVC 5

Pour changer la version de **JDK** et **Dynamic Web Module**

- Faire un clic droit sur le projet et aller dans `Properties`
- Chercher `Java Build Path` puis aller dans `Libraries` et cliquer sur `Edit` pour choisir la version de JDK et enfin valider
- Chercher `Project Facets` puis choisir la version `13` pour `Java` et `4.0` pour `Dynamic Web Module`
- Valider en cliquant sur `Apply and Close`

Spring MVC 5

Pour changer la version de **JDK** et **Dynamic Web Module**

- Faire un clic droit sur le projet et aller dans `Properties`
- Chercher `Java Build Path` puis aller dans `Libraries` et cliquer sur `Edit` pour choisir la version de JDK et enfin valider
- Chercher `Project Facets` puis choisir la version `13` pour `Java` et `4.0` pour `Dynamic Web Module`
- Valider en cliquant sur `Apply and Close`

Relancer le projet