



Spring MVC – set up

Environnement

- Normalement vous avez déjà :
 - Apache Tomcat
 - Eclipse JEE (passez en perspective JEE)
 - Eclipse et tomcat connectés



Configurer Spring MVC

Configurer dans WEB-INF/web.xml :

- déclarer le Dispatcher Servlet
- spécifier une URL d'écoute pour Dispatcher Servlet

Configurer dans WEB-INF/spring-mvc-demo-servlet.xml :

- ajouter component scanning
- Ajouter les fonction conversion, formater et valider
- Ajouter le View Resolver (détermine l'emplacement des vues dans notre application : prefix+vue+suffix)

Créer un projet java (classic)

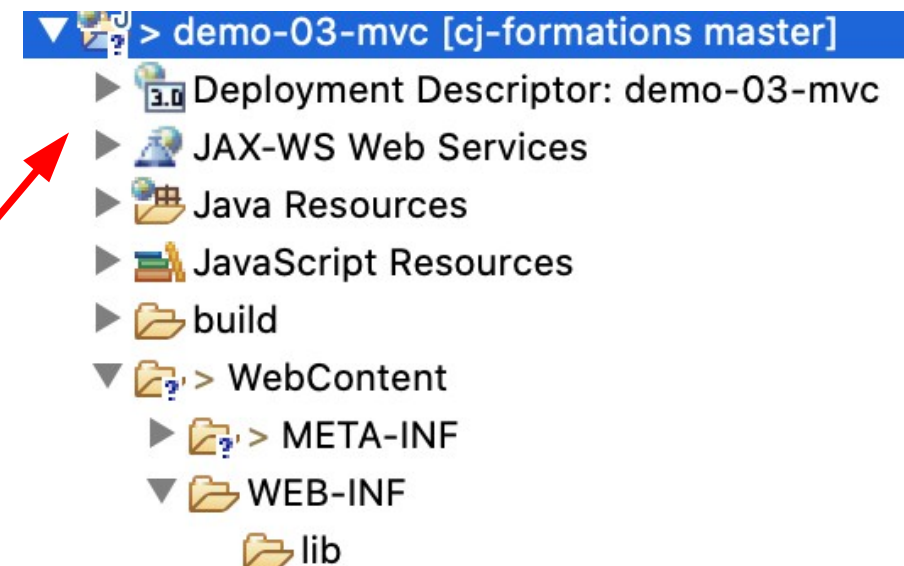
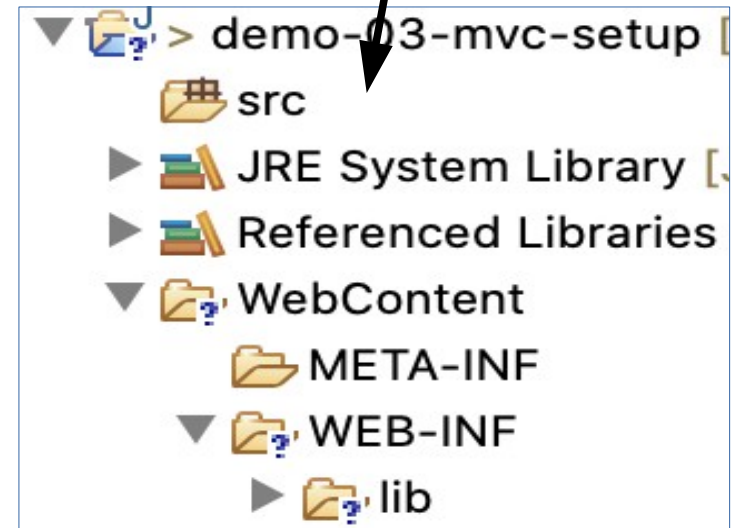
• **Démo – TD** Créez un projet JEE

Nb les répertoires:

- /WebContent
- /WebContent/WEB-INF
- /WebContent/META-INF
- /WebContent/WEB-INF/lib

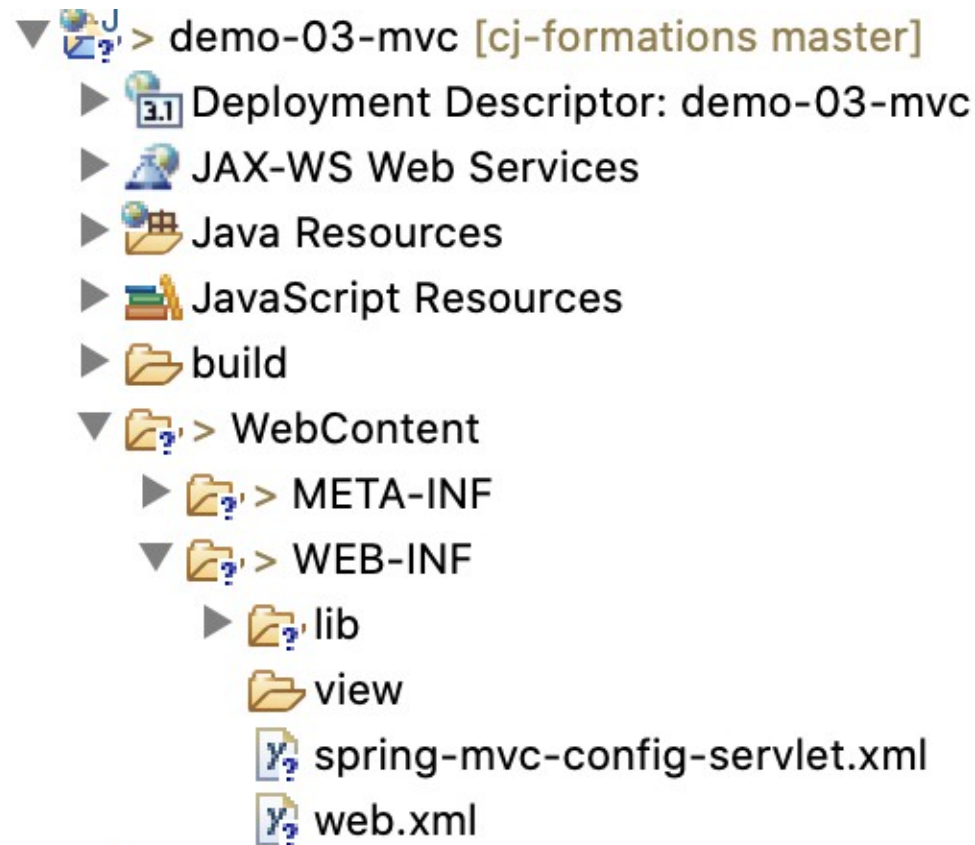
Placez les jars spring dans /lib
Et ajoutez-les au ClassPath

- new Dynamic Web Project)



Classpath automatique

Créez les fichiers xml dans WEB-INF, créez le dossier "view"



Web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd"
id="WebApp_ID" version="3.1">
  <display-name>spring-mvc-demo</display-name>

  <!-- Step 1: déclarer le Dispatcher Servlet -->
  <servlet>
    <servlet-name>dispatcher</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-servlet.xml</param-value>
    </init-param>

    <load-on-startup>1</load-on-startup>
  </servlet>

  <!-- Step 2: spécifier une URL d'écoute -->
  <servlet-mapping>
    <servlet-name>dispatcher</servlet-name>
    <url-pattern>/</url-pattern>
  </servlet-mapping>

  <absolute-ordering/>

</web-app>
```

Spring-mvc-config-servlet.xml

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

<!-- Step 3: Add support for component scanning -->
<context:component-scan base-package="com.tld" />

<!-- Step 4: Add support for conversion, formatting and validation support -->
<mvc:annotation-driven/>

<!-- Step 5: Define Spring MVC view resolver -->
<bean
class="org.springframework.web.servlet.view.InternalResourceViewResolver">
<property name="prefix" value="/WEB-INF/view/" />
<property name="suffix" value=".jsp" />
</bean>

</beans>
```


Il faut que cet emplacement existe

Jars additionnels

En plus des jars de spring ajoutez les jars suivants dans le dossier lib:

- ~~commons logging 1.2.jar~~
- javax.servlet.jsp.jstl-1.2.1.jar
- javax.servlet.jsp.jstl-api-1.2.1.jar

Seulement les deux jars relatifs à jstl



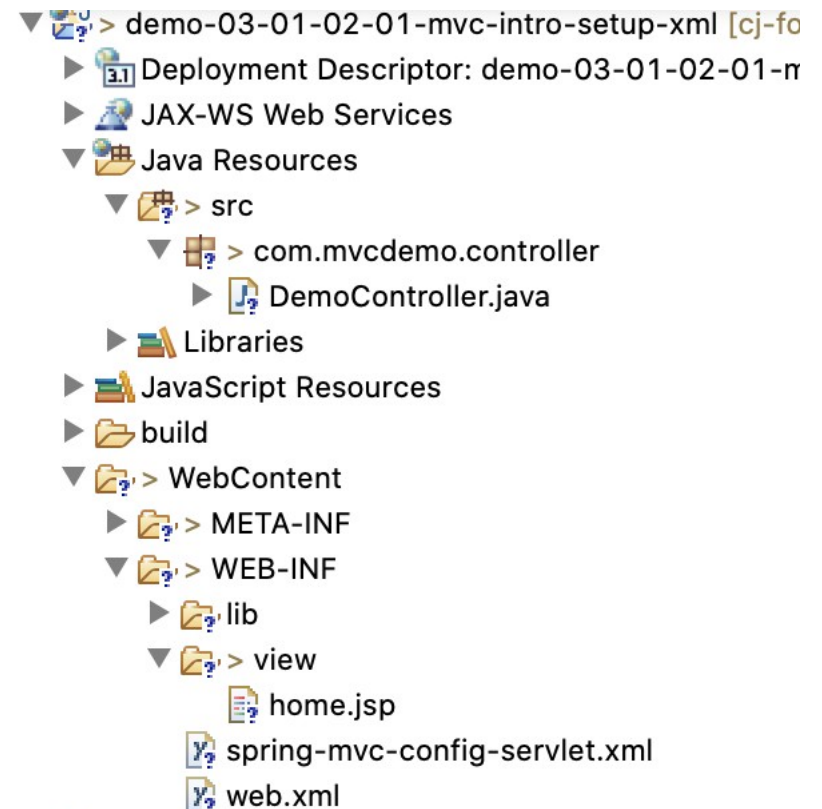
**Si on teste le démarrage
maintenant ,
le serveur va démarrer sans
erreur, et chercher à afficher la
page :
localhost:8080/appName/
...
mais aucune page n'est prévue**

Ajoutons un Contrôleur et une vue

```
@Controller
public class DemoController {

    @GetMapping("/")
    public String showHome() {
        System.out.println("home is responding ");
        return "home";
    }
}
```

```
<html>
<body>
Welcome to the spring body index home page !
</body>
</html>
```



Testons l'application et vérifions que cette page web s'affiche au démarrage de notre application

Web.xml

```
<servlet>
  name : dispatcher
  class : DispatcherServlet
  param-name:contextConfiguration
  param-value : spring-mvc...xml
  load-on-startup 1
</servlet> ...
<url-pattern /
...
```

spring-mvc-config-servlet.xml

```
component-scan :base-package
mvc :annotation-driven
bean class=...ViewResolver prefix & suffix
```

Synthèse :

get : <http://localhost:8080/demo-mvc/>

dispatcher

Mappings : /beta

BetaController

AlphaController

mapping: /

DemoController

"home"

View Resolver

/WEB-INF/view/home.jsp

home.jsp

Response html



FAQ

Configure Spring Dispatcher Servlet
uniquement en utilisant du code Java(no xml)

→ `demo-03-01-02-02-mvc-intro-setup-java`

Nous avons atteint un premier palier de spring MVC

- Si votre page s'affiche
- Et que vous pouvez constater dans les logs de tomcat que le dispatcherServlet est correctement initialisé au démarrage

```
Tomcat v9.0 Server at localhost [Apache Tomcat] /Library/Java/JavaVirtualMachines/openjdk-11.0.2.jdk/Contents/Home
INFO: Démarrage du service [Catalina]
juil. 19, 2020 6:39:40 PM org.apache.catalina.core.StandardEngine startInternal
INFO: Démarrage du moteur de Servlets : [Apache Tomcat/9.0.37]
juil. 19, 2020 6:39:41 PM org.apache.jasper.servlet.TldScanner scanJars
INFO: Au moins un fichier JAR a été analysé pour trouver des TLDs mais il n'en contenai
juil. 19, 2020 6:39:41 PM org.apache.catalina.core.ApplicationContext log
INFO: Initializing Spring DispatcherServlet 'dispatcher'
juil. 19, 2020 6:39:41 PM org.springframework.web.servlet.FrameworkServlet initServlet
INFO: Initializing Servlet 'dispatcher'
juil. 19, 2020 6:39:43 PM org.springframework.web.servlet.FrameworkServlet initServlet
INFO: Completed initialization in 1315 ms
juil. 19, 2020 6:39:43 PM org.apache.coyote.AbstractProtocol start
INFO: Démarrage du gestionnaire de protocole ["http-nio-8080"]
juil. 19, 2020 6:39:43 PM org.apache.catalina.startup.Catalina start
INFO: Le démarrage du serveur a pris [2413] millisecondes
```