

JSF

JAVA SERVER FACES



Mohamed Youssfi

med@youssfi.net

ENSET, Université Hassan II Mohammeda Casablanca

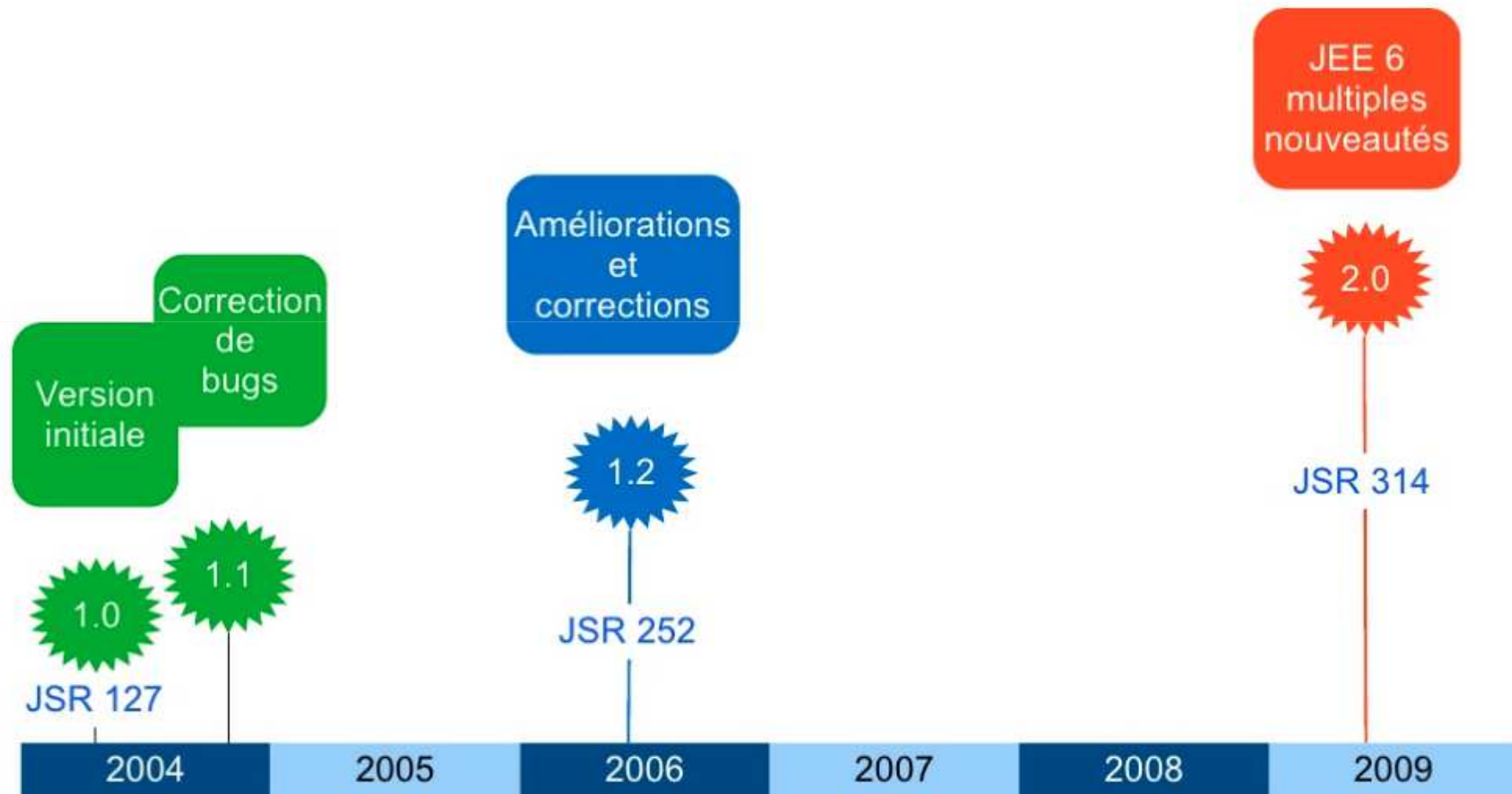
JSF

- Java Server Faces (JSF) est une technologie dont le but est de proposer un framework qui facilite et standardise le développement d'applications web avec Java.
- JSF est un Framework orienté composants
- Son développement a tenu en compte des différentes expériences acquises lors de l'utilisation des technologies standards pour le développement d'applications web
 - Servlet, JSP, JSTL
 - et de différents frameworks (Struts, Spring MVC).

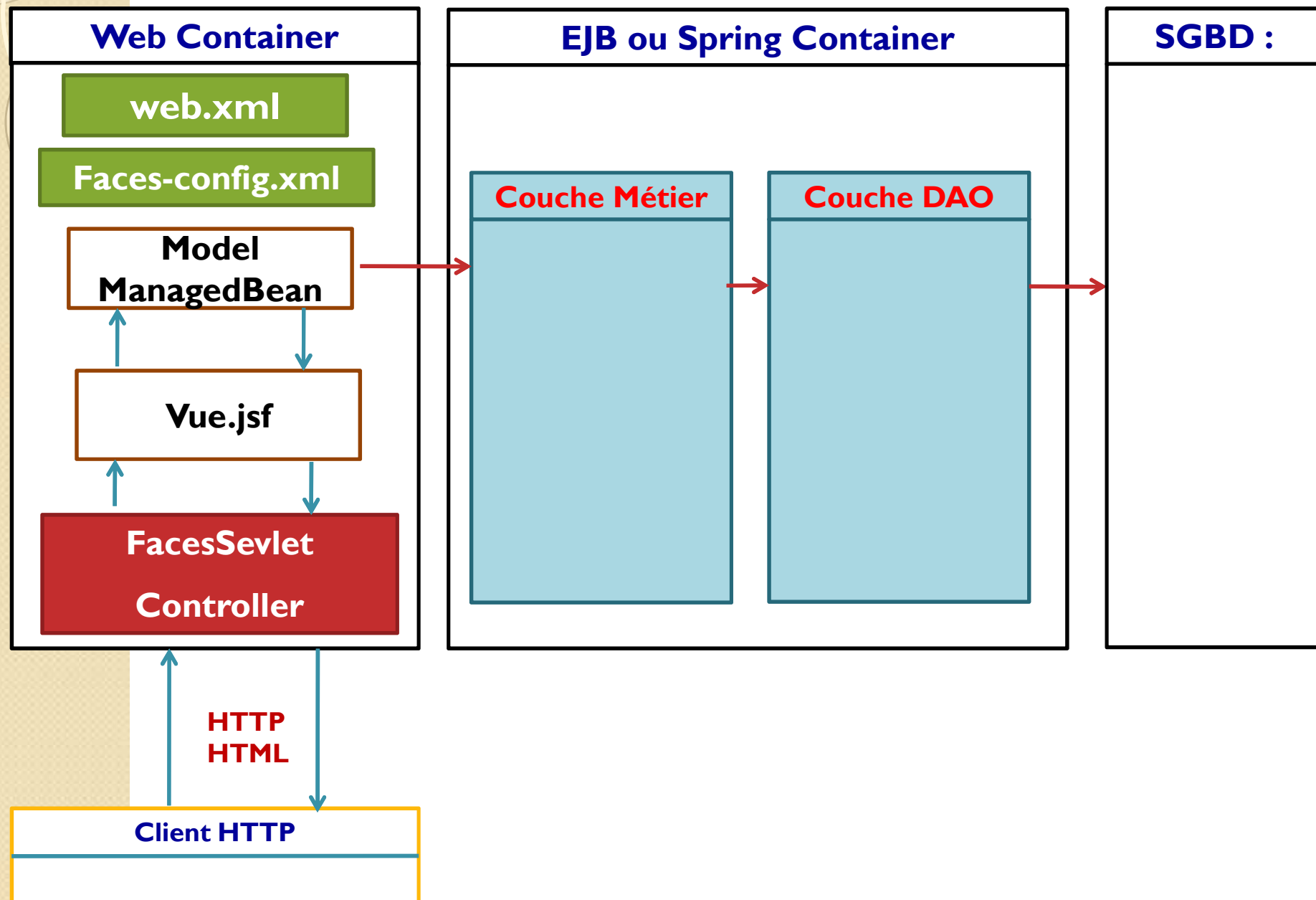
JSF

- L'objectif de JSF est de :
 - fournir un standard JEE spécifié dans une JSR pour le développement des IHM web riches
 - Maximiser la productivité des applications web
 - Fournir des fonctionnalités récurrentes et avancées (Validations, Conversion, Ajax ...)
 - Masquer la complexité

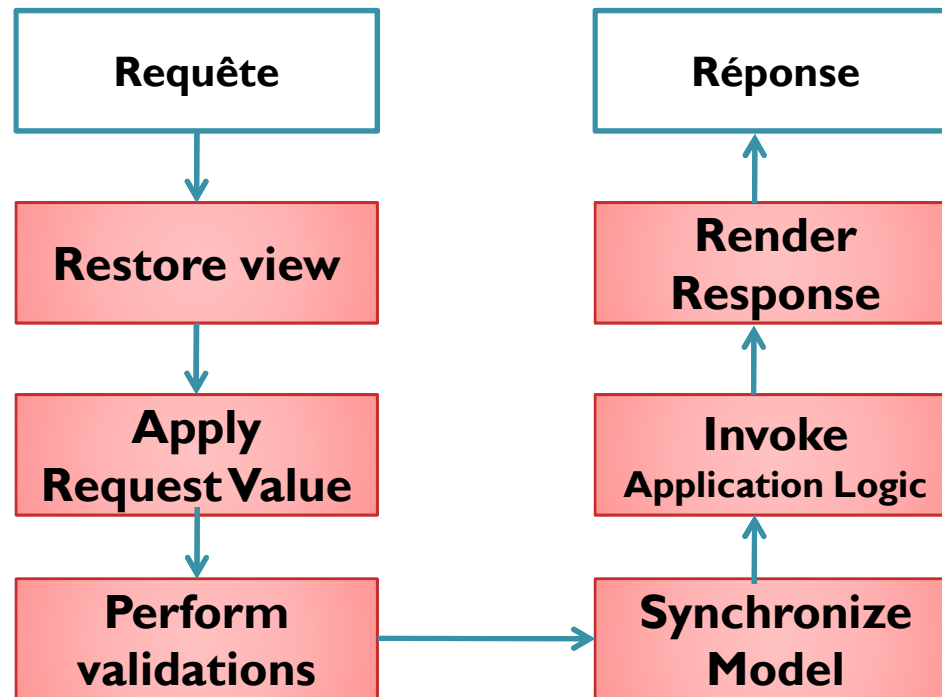
Historique



Architecture



Cycle de vie



Cycle de vie

- **Restore view ou Reconstruct Component Tree :**
 - Cette première phase permet au serveur de recréer l'arborescence des composants qui composent la page.
 - Cette arborescence est stockée dans un objet de type FacesContext et sera utilisée tout au long du traitement de la requête.

Cycle de vie

- **Apply Request Value :**
 - Dans cette étape, les valeurs des données sont extraites de la requête HTTP pour chaque composant et sont stockées dans leur composant respectif dans le FaceContext.
 - Durant cette phase des opérations de conversions sont réalisées pour permettre de transformer les valeurs stockées sous forme de chaîne de caractères dans la requête http en un type utilisé pour le stockage des données.

Cycle de vie

- **Perform validations :**
 - Une fois les données extraites et converties, il est possible de procéder à leur validation en appliquant les validators enregistrés auprès de chaque composant.
 - Les éventuelles erreurs de conversions sont stockées dans le FaceContext.
 - Dans ce cas, l'étape suivante est directement « **Render Response** » pour permettre de réafficher la page avec les valeurs saisies et afficher les erreurs

Cycle de vie

- **Synchronize Model ou update model values :**
 - Cette étape permet de stocker dans les composants du FaceContext leur valeur locale validée respective.
 - Les éventuelles erreurs de conversions sont stockées dans le FaceContext.
 - Dans ce cas, l'étape suivante est directement « **Render Response** » pour permettre de réafficher la page avec les valeurs saisies et afficher les erreurs

Cycle de vie

- **Invoke Application Logic :**
 - Dans cette étape, le ou les événements émis dans la page sont traités.
 - Cette phase doit permettre de déterminer quelle sera la page résultat qui sera renvoyée dans la réponse en utilisant les règles de navigation définie dans l'application.
 - L'arborescence des composants de cette page est créée.
- **Render Response :**
 - Cette étape se charge de créer le rendu de la page de la réponse.

Web.xml

```
<servlet>
  <servlet-name>Faces Servlet</servlet-name>
  <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
  <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
  <servlet-name>Faces Servlet</servlet-name>
  <url-pattern>*.xhtml</url-pattern>
</servlet-mapping>
```

Maven dependencies : pom.xml

```
<dependency>  
  <groupId>com.sun.faces</groupId>  
  <artifactId>jsf-api</artifactId>  
  <version>2.2.1</version>  
</dependency>  
<dependency>  
  <groupId>com.sun.faces</groupId>  
  <artifactId>jsf-impl</artifactId>  
  <version>2.2.1</version>  
</dependency>
```

Maven dependencies : pom.xml

```
<dependency>
  <groupId>org.richfaces.ui</groupId>
  <artifactId>richfaces-components-api</artifactId>
  <version>4.2.2.Final</version>
</dependency>
<dependency>
  <groupId>org.richfaces.ui</groupId>
  <artifactId>richfaces-components-ui</artifactId>
  <version>4.2.2.Final</version>
</dependency>
```

Maven dependencies : pom.xml

```
<dependency>
  <groupId>org.richfaces.core</groupId>
  <artifactId>richfaces-core-api</artifactId>
  <version>4.2.2.Final</version>
</dependency>
<dependency>
  <groupId>org.richfaces.core</groupId>
  <artifactId>richfaces-core-impl</artifactId>
  <version>4.2.2.Final</version>
</dependency>
```

faces-config.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<faces-config
  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-facesconfig_2_2.xsd"
  version="2.2">

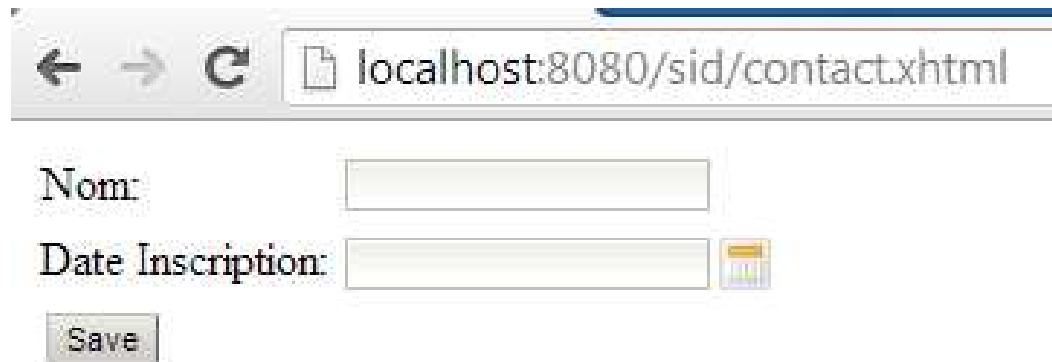
</faces-config>
```


Exemple de Managed Bean

```
package org.miage.sid.jsfmb;
import java.util.Date; import javax.faces.bean.ManagedBean;
import javax.faces.bean.RequestScoped;
@ManagedBean(name="contactBean")
@RequestScoped
public class ContactBean {
    private String nom;
    private Date dateNaissance;
    public String saveContact(){
        // Traitement
        System.out.println(nom);
        System.out.println(dateNaissance);
        return "success";
    }
    // Getters et Setters
}
```

Exemple de vue JSF : contact.xhtml

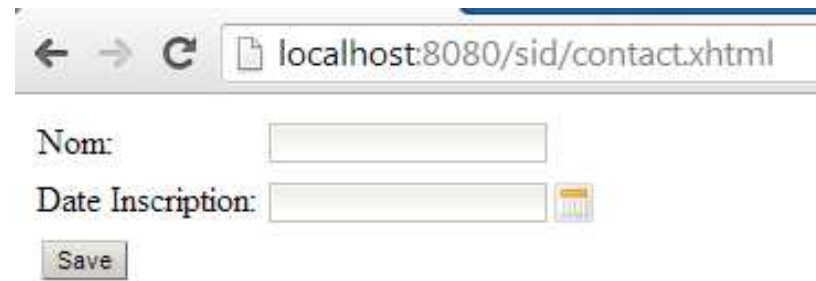
```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml" xmlns:f="http://xmlns.jcp.org/jsf/core"
      xmlns:h="http://xmlns.jcp.org/jsf/html" xmlns:r="http://richfaces.org/rich">
<h:head>
  <meta charset="UTF-8"/>
  <title>Contact</title>
</h:head>
<h:body>
  <f:view>
```



A screenshot of a web browser window displaying a contact form. The address bar shows the URL 'localhost:8080/sid/contact.xhtml'. The form contains two input fields: 'Nom:' and 'Date Inscription:'. The 'Date Inscription:' field has a calendar icon to its right. Below the input fields is a 'Save' button.

Exemple de vue JSF : contact.xhtml

```
<h:form>
  <h:panelGrid columns="3">
    <h:outputText value="Nom:"/>
    <h:inputText id="nom" value="#{contactBean.nom}" label="Nom:">
      <f:validateLength minimum="4" maximum="12"/>
    </h:inputText>
    <h:message for="nom" errorStyle="color:red"/>
    <h:outputText value="Date Inscription:"/>
    <r:calendar id="dateNaissance" value="#{contactBean.dateNaissance}"
      popup="true" cellWidth="24px" cellHeight="22px"
      style="width:200px"/>
    <h:message for="dateNaissance" errorStyle="color:red"/>
    <h:commandButton action="#{contactBean.saveContact}" value="Save"/>
  </h:panelGrid>
</h:form>
</f:view>
</h:body>
</html>
```



← → ↻ 📄 localhost:8080/sid/contact.xhtml

Nom:

Date Inscription: 📅

Les Composants JSF

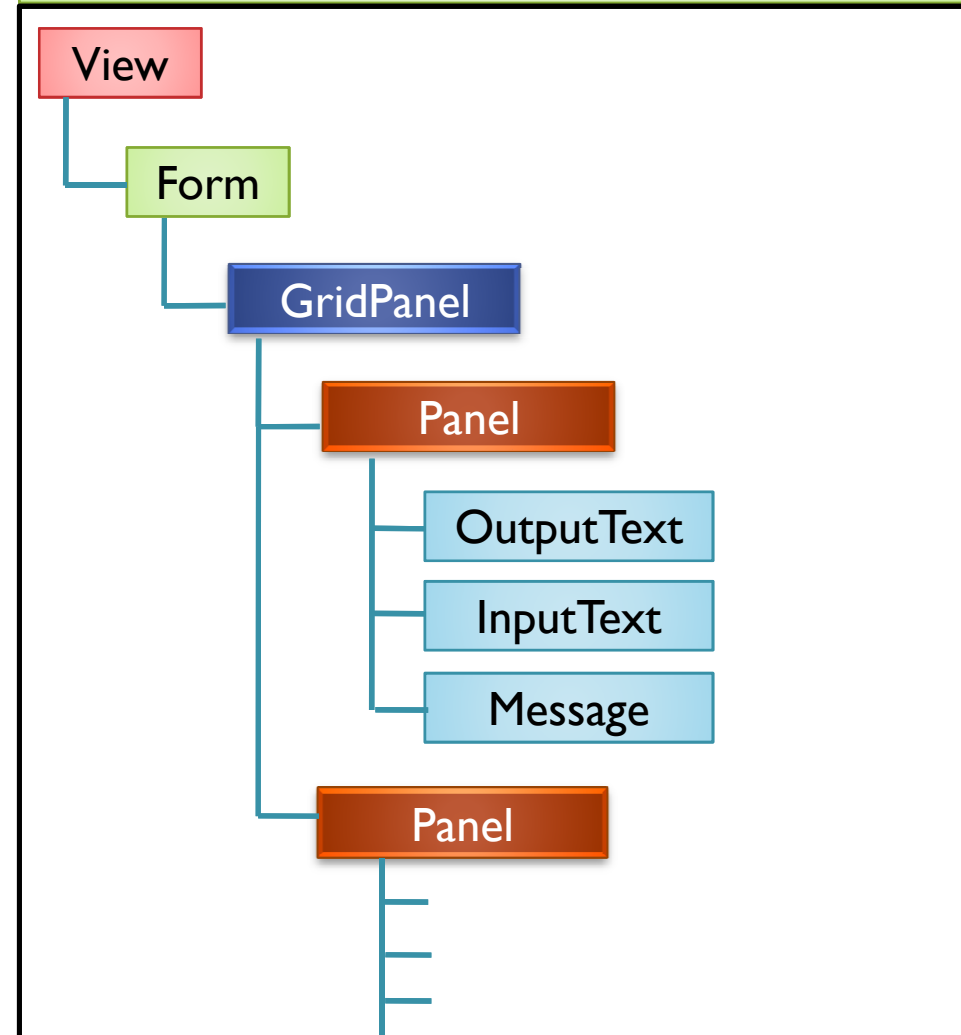
Coté Client

← → ↻ localhost:8080/sid/contact.xhtml

Nom:

Date Inscription:

Coté Serveur





INTÉGRATION AVEC SPRING

Application I

- On souhaite créer une application qui permet de gérer des produits. Chaque produit est défini par sa référence (de type String), sa désignation, son prix et sa quantité. L'application doit permettre les opérations suivantes :
 - Ajouter un nouveau produit
 - Consulter tous les produits
 - Consulter les produits dont le nom contient un mot clé.
 - Consulter un produit
 - Supprimer un produit
 - Mettre à jour un produit.
- Cette application se compose de trois couches DAO, Métier et Présentation.
- Elle doit être fermée à la modification et ouverte à l'extension.
- L'injection des dépendances sera effectuée en utilisant Spring IOC.
- Nous allons définir deux implémentations de la couche DAO.
 - Une implémentation qui gère les produits qui sont stockés dans une liste de type HashMap.
 - Dans la deuxième implémentation, nous allons supposer que les produits sont stockés dans une base de données de type MySQL.

Gestion des produits

Insert title here

http://localhost:8080/sid/index.xhtml

REF

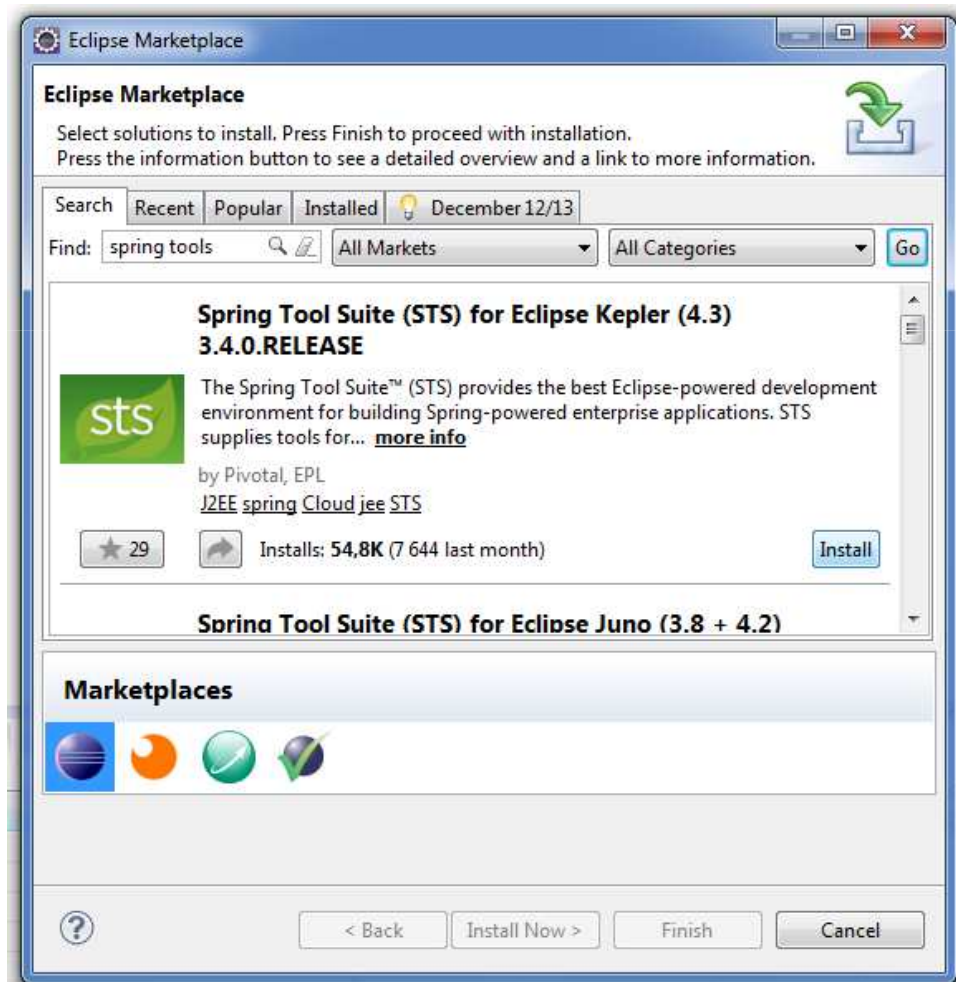
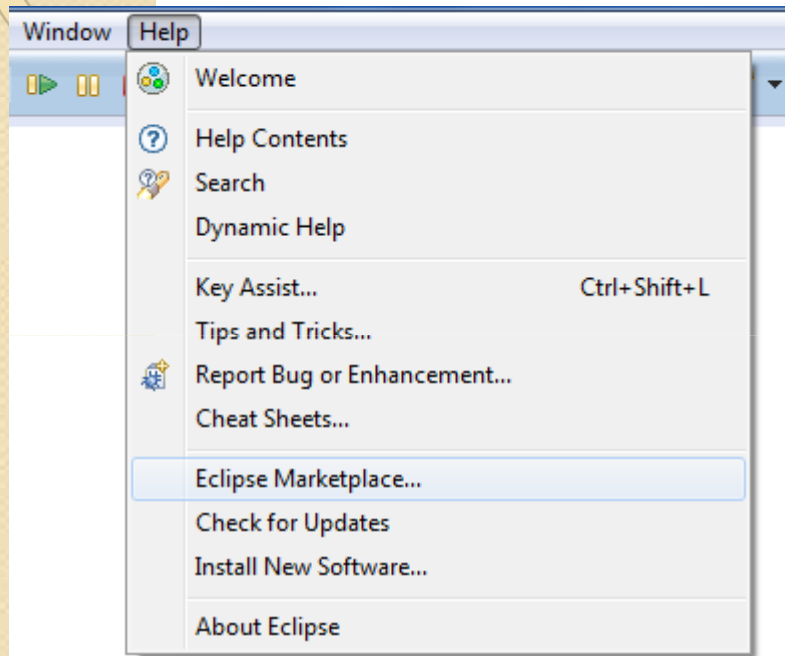
DESIGNATION

PRIX

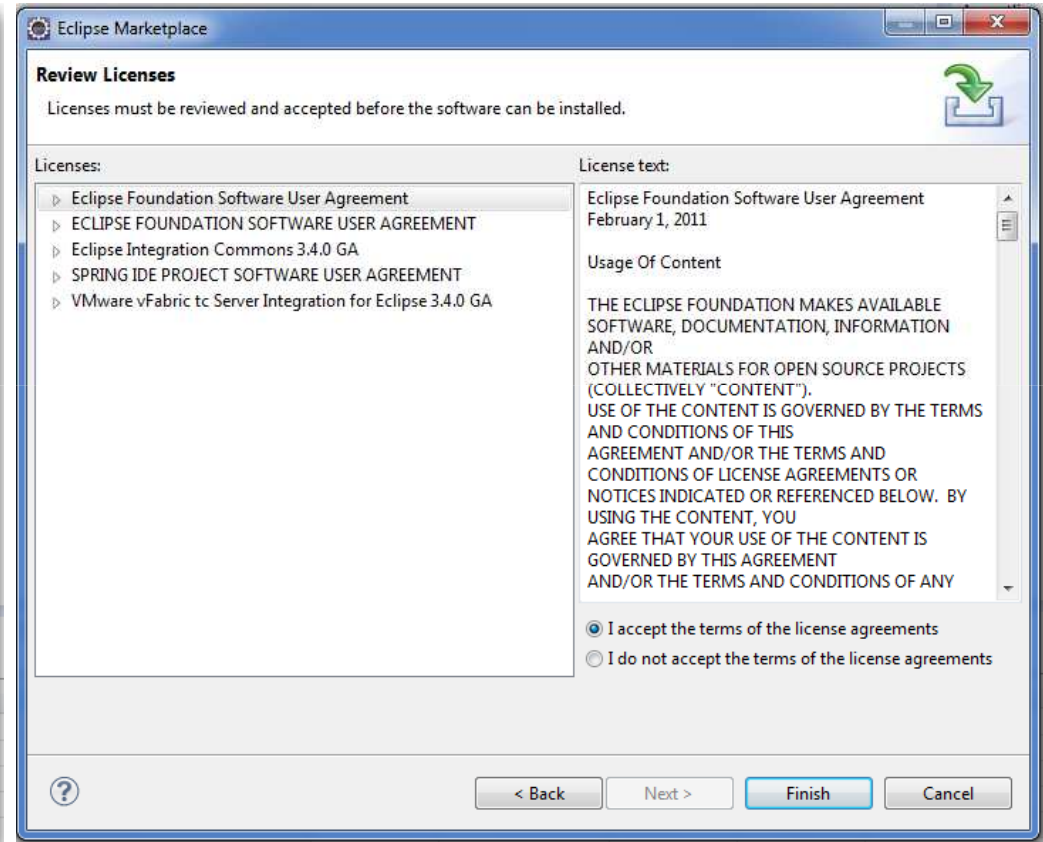
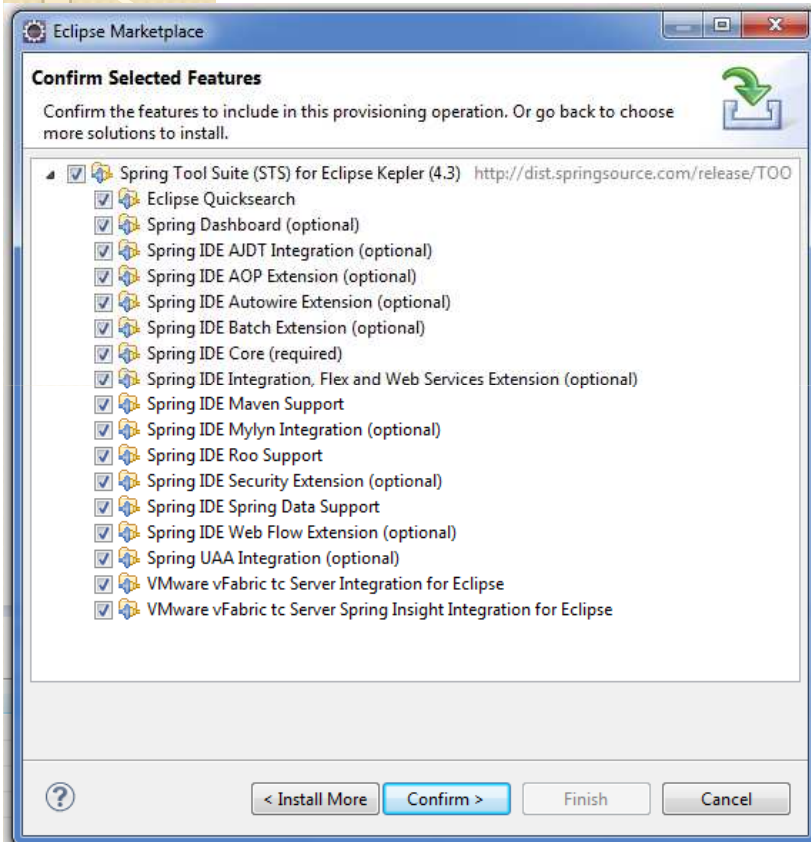
QUANTITE

REF	DESIGNATION	PRIX	QUANTITE		
AEA	Imprimante 543	6000.0	10	Supp	Edit
AI321	Smart phone LG	5432.0	14	Supp	Edit
HP654	Ordinateur HP564	8000.0	12	Supp	Edit

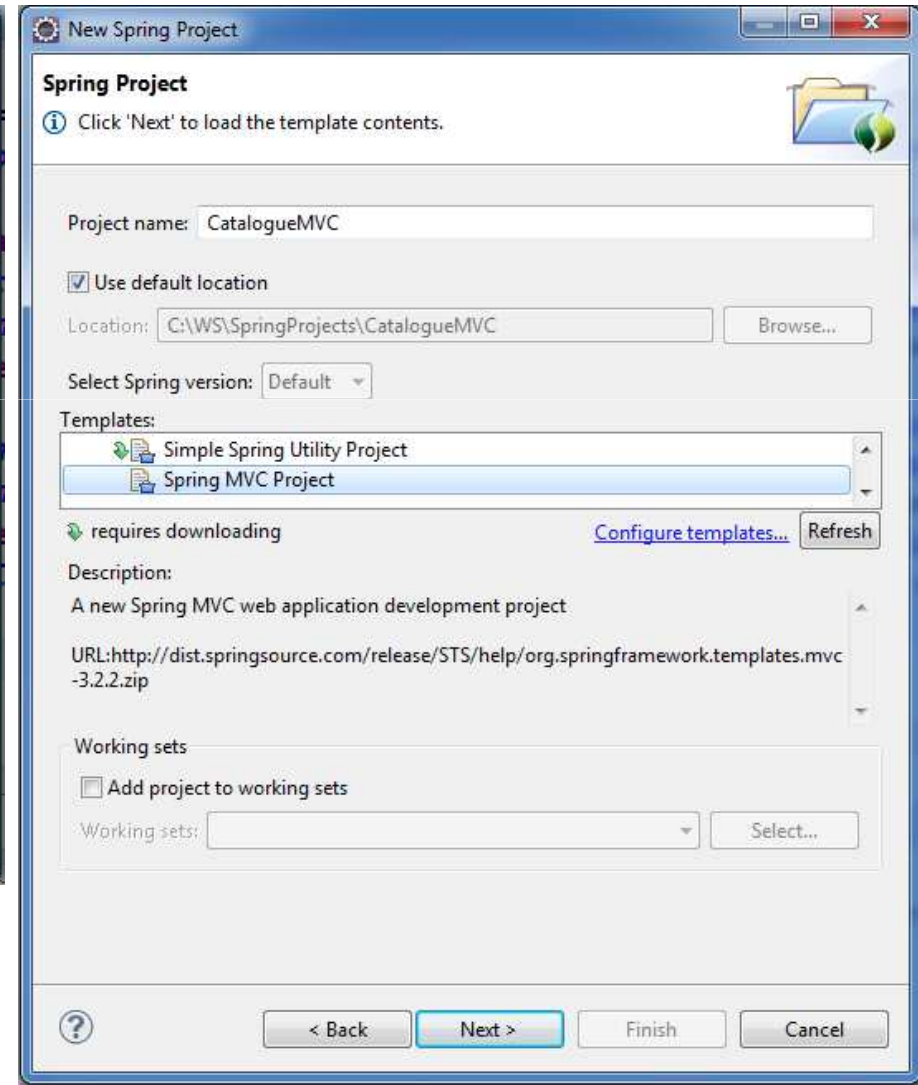
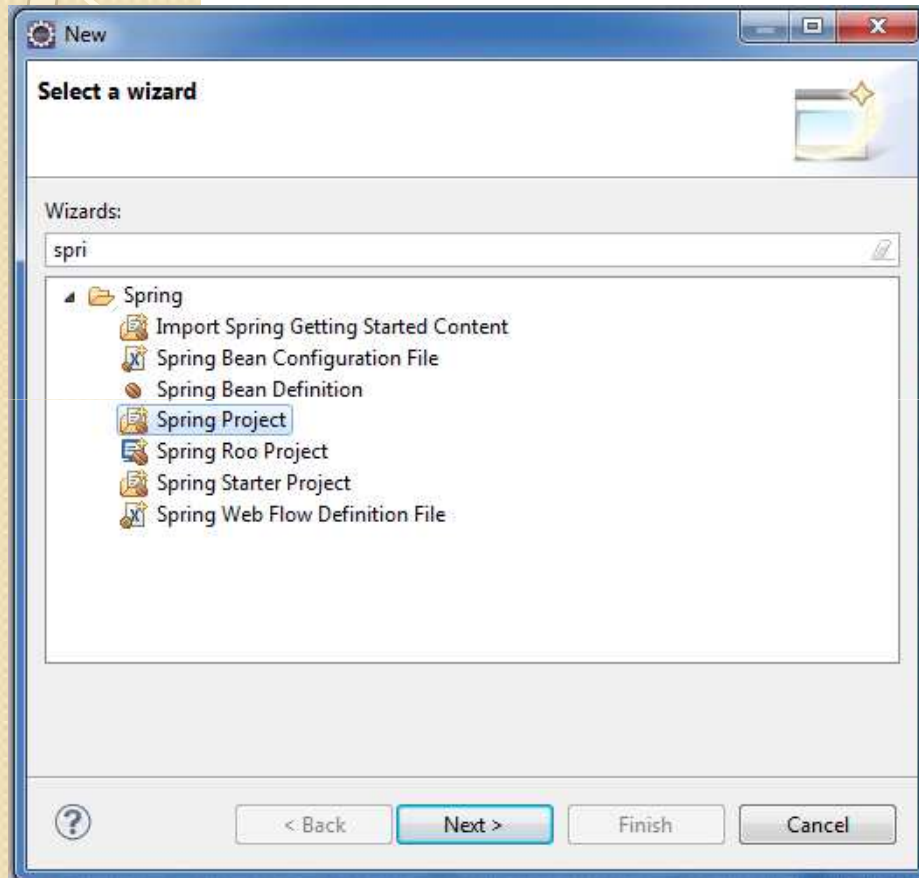
Installation du plugin : spring tools pour eclipse



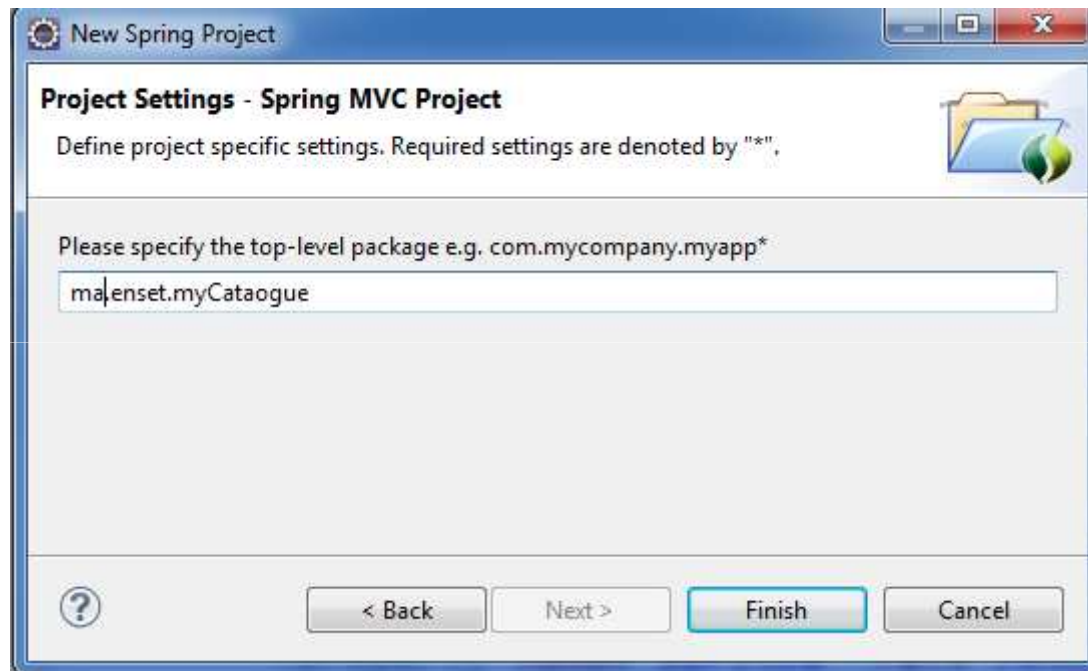
Installation du plugin : spring tools pour eclipse



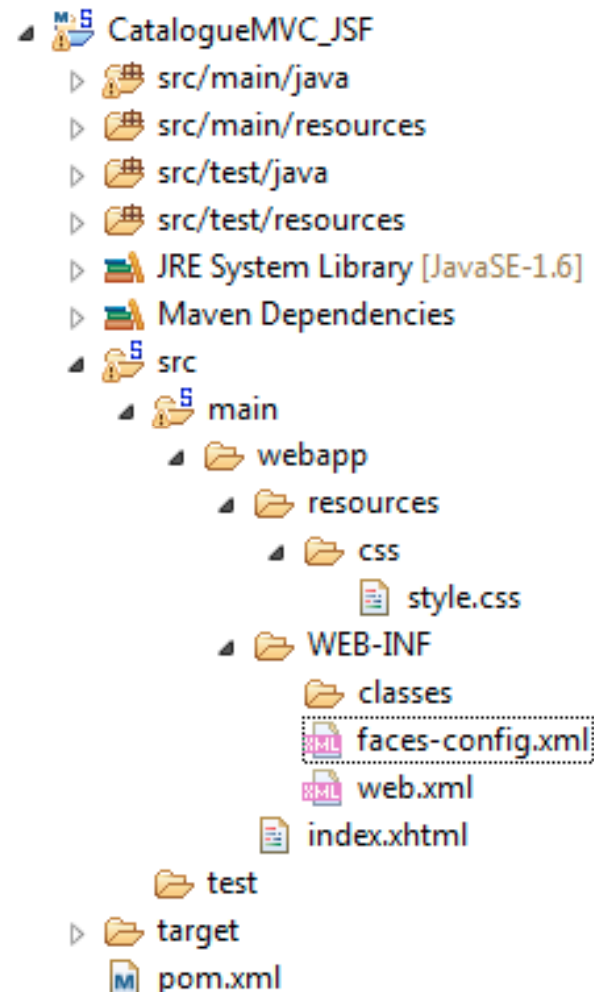
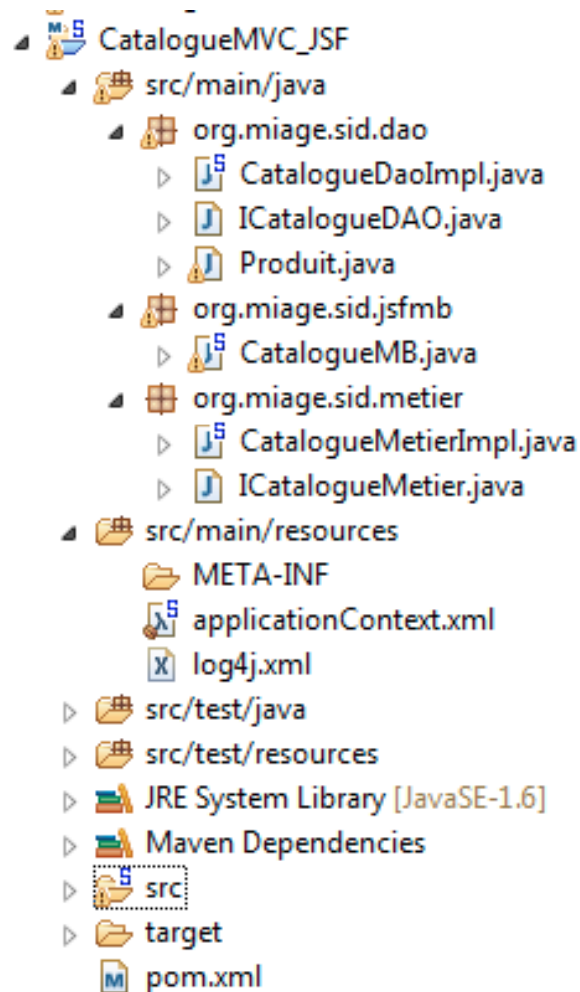
Création d'un projet Spring



Création d'un projet Spring



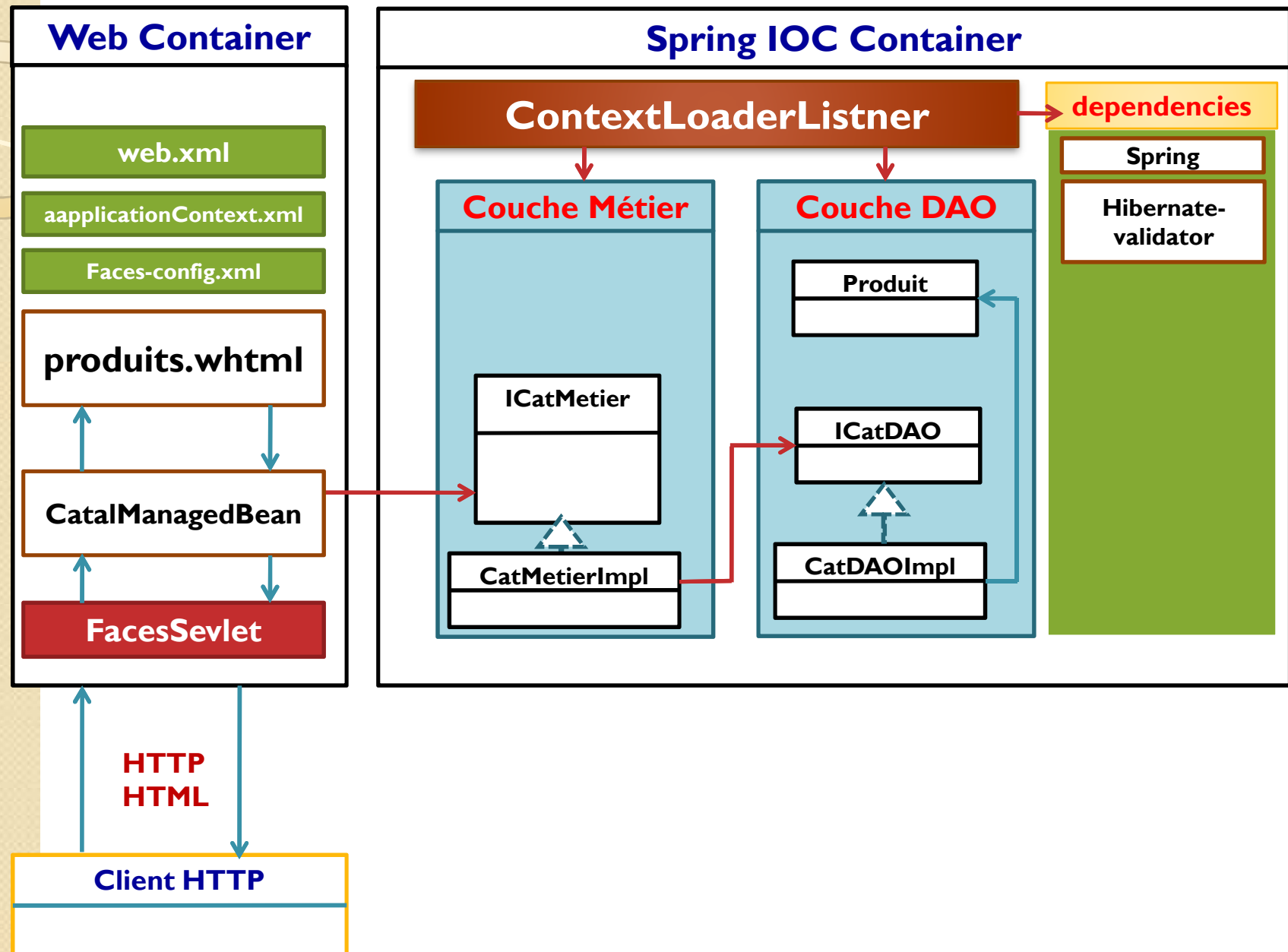
Structure du projet



Dépendances

- ▲ Maven Dependencies
- ▶ spring-context-3.1.1.RELEASE.jar - C:\Users\youssfi\.m2\repository\org\springframework\spring-context\3.1.1.RELEASE
 - ▶ spring-aop-3.1.1.RELEASE.jar - C:\Users\youssfi\.m2\repository\org\springframework\spring-aop\3.1.1.RELEASE
 - ▶ aopalliance-1.0.jar - C:\Users\youssfi\.m2\repository\aopalliance\aopalliance\1.0
 - ▶ spring-beans-3.1.1.RELEASE.jar - C:\Users\youssfi\.m2\repository\org\springframework\spring-beans\3.1.1.RELEASE
 - ▶ spring-core-3.1.1.RELEASE.jar - C:\Users\youssfi\.m2\repository\org\springframework\spring-core\3.1.1.RELEASE
 - ▶ spring-expression-3.1.1.RELEASE.jar - C:\Users\youssfi\.m2\repository\org\springframework\spring-expression\3.1.1.RELEASE
 - ▶ spring-asm-3.1.1.RELEASE.jar - C:\Users\youssfi\.m2\repository\org\springframework\spring-asm\3.1.1.RELEASE
 - ▶ spring-webmvc-3.1.1.RELEASE.jar - C:\Users\youssfi\.m2\repository\org\springframework\spring-webmvc\3.1.1.RELEASE
 - ▶ spring-context-support-3.1.1.RELEASE.jar - C:\Users\youssfi\.m2\repository\org\springframework\spring-context-support\3.1.1.RELEASE
 - ▶ spring-web-3.1.1.RELEASE.jar - C:\Users\youssfi\.m2\repository\org\springframework\spring-web\3.1.1.RELEASE
 - ▶ aspectjrt-1.6.10.jar - C:\Users\youssfi\.m2\repository\org\aspectj\aspectjrt\1.6.10
 - ▶ slf4j-api-1.6.6.jar - C:\Users\youssfi\.m2\repository\org\slf4j\slf4j-api\1.6.6
 - ▶ jcl-over-slf4j-1.6.6.jar - C:\Users\youssfi\.m2\repository\org\slf4j\jcl-over-slf4j\1.6.6
 - ▶ slf4j-log4j12-1.6.6.jar - C:\Users\youssfi\.m2\repository\org\slf4j\slf4j-log4j12\1.6.6
 - ▶ log4j-1.2.15.jar - C:\Users\youssfi\.m2\repository\log4j\log4j\1.2.15
 - ▶ javax.inject-1.jar - C:\Users\youssfi\.m2\repository\javax\inject\javax.inject\1
 - ▶ servlet-api-2.5.jar - C:\Users\youssfi\.m2\repository\javax\servlet\servlet-api\2.5
 - ▶ jsp-api-2.1.jar - C:\Users\youssfi\.m2\repository\javax\servlet\jsp\jsp-api\2.1
 - ▶ jstl-1.2.jar - C:\Users\youssfi\.m2\repository\javax\servlet\jstl\1.2
 - ▶ junit-4.7.jar - C:\Users\youssfi\.m2\repository\junit\junit\4.7
 - ▶ hibernate-validator-5.0.0.Final.jar - C:\Users\youssfi\.m2\repository\org\hibernate\hibernate-validator\5.0.0.Final
 - ▶ jboss-logging-3.1.1.GA.jar - C:\Users\youssfi\.m2\repository\org\jboss\logging\jboss-logging\3.1.1.GA
 - ▶ classmate-0.8.0.jar - C:\Users\youssfi\.m2\repository\com\fastxml\classmate\0.8.0
 - ▶ javax.el-2.2.4.jar - C:\Users\youssfi\.m2\repository\org\glassfish\web\javax.el\2.2.4
 - ▶ javax.el-api-2.2.4.jar - C:\Users\youssfi\.m2\repository\javax\el\javax.el-api\2.2.4
 - ▶ validation-api-1.1.0.Final.jar - C:\Users\youssfi\.m2\repository\javax\validation\validation-api\1.1.0.Final
 - ▶ jsf-api-2.2.1.jar - C:\Users\youssfi\.m2\repository\com\sun\faces\jsf-api\2.2.1
 - ▶ jsf-impl-2.2.1.jar - C:\Users\youssfi\.m2\repository\com\sun\faces\jsf-impl\2.2.1

Architecture technique



Maven dependencies : Spring

pom.xml

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

Maven Dependencies

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


Maven Dependencies

```
<!-- AspectJ -->
<dependency>
  <groupId>org.aspectj</groupId>
  <artifactId>aspectjrt</artifactId>
  <version>${org.aspectj-version}</version>
</dependency>
<!-- Logging -->
<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4j-api</artifactId>
  <version>${org.slf4j-version}</version>
</dependency>
<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>jcl-over-slf4j</artifactId>
  <version>${org.slf4j-version}</version>
  <scope>runtime</scope>
</dependency>
<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4j-log4j12</artifactId>
  <version>${org.slf4j-version}</version>
  <scope>runtime</scope>
</dependency>
```

Maven Dependencies

```
<dependency>
  <groupId>log4j</groupId>
  <artifactId>log4j</artifactId>
  <version>1.2.15</version>
  <exclusions>
    <exclusion>
      <groupId>javax.mail</groupId>
      <artifactId>mail</artifactId>
    </exclusion>
    <exclusion>
      <groupId>javax.jms</groupId>
      <artifactId>jms</artifactId>
    </exclusion>
    <exclusion>
      <groupId>com.sun.jdmk</groupId>
      <artifactId>jmxtools</artifactId>
    </exclusion>
    <exclusion>
      <groupId>com.sun.jmx</groupId>
      <artifactId>jmxri</artifactId>
    </exclusion>
  </exclusions>
  <scope>runtime</scope>
</dependency>
```

Maven Dependencies

```
<!-- @Inject -->
<dependency>
  <groupId>javax.inject</groupId>
  <artifactId>javax.inject</artifactId>
  <version>1</version>
</dependency>

<!-- Servlet -->
<dependency>
  <groupId>javax.servlet</groupId>
  <artifactId>servlet-api</artifactId>
  <version>2.5</version>
  <scope>provided</scope>
</dependency>
<dependency>
  <groupId>javax.servlet.jsp</groupId>
  <artifactId>jsp-api</artifactId>
  <version>2.1</version>
  <scope>provided</scope>
</dependency>
<dependency>
  <groupId>javax.servlet</groupId>
  <artifactId>jstl</artifactId>
  <version>1.2</version>
</dependency>
```

Maven Dependencies

```
<!-- Test -->
<dependency>
  <groupId>junit</groupId>
  <artifactId>junit</artifactId>
  <version>4.7</version>
  <scope>test</scope>
</dependency>
<!-- Hibernate Validator -->
<dependency>
  <groupId>org.hibernate</groupId>
  <artifactId>hibernate-validator</artifactId>
  <version>5.0.0.Final</version>
</dependency>
<dependency>
  <groupId>javax.validation</groupId>
  <artifactId>validation-api</artifactId>
  <version>1.1.0.Final</version>
</dependency>
```

Maven Dependencies

```
<!-- JSF -->
```

```
<dependency>
```

```
  <groupId>com.sun.faces</groupId>
```

```
  <artifactId>jsf-api</artifactId>
```

```
  <version>2.2.1</version>
```

```
</dependency>
```

```
<dependency>
```

```
  <groupId>com.sun.faces</groupId>
```

```
  <artifactId>jsf-impl</artifactId>
```

```
  <version>2.2.1</version>
```

```
</dependency>
```



COUCHE DAO

Entité Produit

```
package org.miage.sid.dao;
import java.io.Serializable;
import javax.validation.constraints.DecimalMin;
import javax.validation.constraints.Size;
import org.hibernate.validator.constraints.NotEmpty;
public class Produit implements Serializable {
    @NotEmpty
    @Size(min=4,max=12)
    private String reference;
    @NotEmpty
    private String designation;
    @DecimalMin(value="100")
    private double prix;
    private int quantite;
    // Constructeurs
    // Getters et Setters
}
```

Interface ICatalogueDAO

```
package org.miage.sid.dao;
import java.util.List;
public interface ICatalogueDAO {
    public void addProduit(Produit p);
    public List<Produit> getAllProduits();
    public List<Produit> getProduits(String mc);
    public Produit getProduit(String reference);
    public void deleteProduit(String refernce);
    public void updateProduit(Produit p);
}
```


Implémentation CatalogueDAOImpl

```
package org.miage.sid.dao;
import java.util.*;
import org.apache.log4j.Logger;
public class CatalogueDaoImpl implements ICatalogueDAO {
    private Map<String, Produit> produits=new HashMap<String, Produit>();
    Logger logger=Logger.getLogger(CatalogueDaoImpl.class);
    @Override
    public void addProduit(Produit p) {
        produits.put(p.getReference(), p);
    }
    @Override
    public List<Produit> getAllProduits() {
        Collection<Produit> prods=produits.values();
        return new ArrayList<Produit>(prods);
    }
}
```

Implémentation CatalogueDAOImpl

```
@Override
public List<Produit> getProduits(String mc) {
    List<Produit> prods=new ArrayList<Produit>();
    for(Produit p:produits.values())
        if(p.getDesignation().indexOf(mc)>=0)
            prods.add(p);
    return prods;
}

@Override
public Produit getProduit(String reference) {
    return produits.get(reference);
}

@Override
public void deleteProduit(String refernce) {
    produits.remove(refernce);
}
```

Implémentation CatalogueDAOImpl

@Override

```
public void updateProduit(Produit p) {  
    produits.put(p.getReference(),p);  
}  
  
public void init(){  
    logger.info("Initialisation du catalogue");  
    this.addProduit(new Produit("HP675","Ordinateur HP", 8000, 5));  
    this.addProduit(new Produit("AEP65","Impriomante AE",760, 80));  
    this.addProduit(new Produit("AT980","Smart Phone GT", 4500, 8));  
}  
}
```



COUCHE METIER

Interface ICatalogueMetier

```
package org.miage.sid.metier;
import java.util.List;
import org.miage.sid.dao.Produit;
public interface ICatalogueMetier {
    public void addProduit(Produit p);
    public List<Produit> getAllProduits();
    public List<Produit> getProduits(String mc);
    public Produit getProduit(String reference);
    public void deleteProduit(String refernce);
    public void updateProduit(Produit p);
}
```

Implémentation CatalogueMetierImpl

```
package org.miage.sid.metier;
import java.util.List; import org.miage.sid.dao.ICatalogueDAO;
import org.miage.sid.dao.Produit;
public class CatalogueMetierImpl implements ICatalogueMetier {
    private ICatalogueDAO dao;
    /*Setter setDao pour l'injection*/
    public void setDao(ICatalogueDAO dao) {
        this.dao = dao;
    }
    @Override
    public void addProduit(Produit p) {
        dao.addProduit(p);
    }
    @Override
    public List<Produit> getAllProduits() {
        return dao.getAllProduits();
    }
}
```

Implémentation CatalogueMetierImpl

```
@Override
public List<Produit> getProduits(String mc) {
    return dao.getProduits(mc);
}

@Override
public Produit getProduit(String reference) {
    return dao.getProduit(reference);
}

@Override
public void deleteProduit(String refernce) {
    dao.deleteProduit(refernce);
}

@Override
public void updateProduit(Produit p) {
    dao.updateProduit(p);
}
}
```



INJECTION DES DEPENDANCES

applicationContext.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"
xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">
  <bean id="dao" class="org.miage.sid.dao.CatalogueDaoImpl" init-method="init"></bean>
  <bean id="metier" class="org.miage.sid.metier.CatalogueMetierImpl">
    <property name="dao" ref="dao"></property>
  </bean>
</beans>
```





COUCHE WEB

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
    http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd">
<!-- The definition of the Root Spring Container shared by all Servlets and Filters -->
<context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>classpath*:applicationContext.xml</param-value>
</context-param>
<!-- Creates the Spring Container shared by all Servlets and Filters -->
<listener>
<listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
</listener>
```

web.xml

```
<servlet>
  <servlet-name>Faces Servlet</servlet-name>
  <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
  <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
  <servlet-name>Faces Servlet</servlet-name>
  <url-pattern>*.xhtml</url-pattern>
</servlet-mapping>
</web-app>
```

faces-config.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<faces-config
  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-facesconfig_2_2.xsd"
  version="2.2">
  <application>
    <el-resolver>
      org.springframework.web.jsf.el.SpringBeanFacesELResolver
    </el-resolver>
  </application>
</faces-config>
```

JSF ManagedBean

```
package org.miage.sid.jsfmb;

import java.util.List;import javax.faces.bean.*;import org.miage.sid.dao.Produit;
import org.miage.sid.metier.ICatalogueMetier;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Component;

@Component
@ManagedBean
@RequestScoped

public class CatalogueMB {
    private Produit produit=new Produit();
    @Autowired
    private ICatalogueMetier metier;
    public List<Produit> getListProduits(){
        return metier.getAllProduits();
    }
    public String saveProduit(){
        metier.addproduit(produit);
        return "sucess";
    }
}
```

JSF ManagedBean

```
public String deleteProduit(String ref){
    metier.deleteProduit(ref);
    return "sucess";
}

public String editProduit(String ref){
    if(metier.getproduit(ref)!=null)
        produit=metier.getproduit(ref);
    return "sucess ";
}

// Getters et Setters
public Produit getProduit() {
    return produit;
}

public void setProduit(Produit produit) {
    this.produit = produit;
}
}
```

Page JSF : index.xhtml

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xmlns:f="http://xmlns.jcp.org/jsf/core"
    xmlns:h="http://xmlns.jcp.org/jsf/html">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
<title>Insert title here</title>
<link rel="stylesheet" type="text/css" href="resources/css/style.css"></link>
</head>
<body>
```


Page JSF : index.xhtml

```
<h:form>
```

```
<div>
```

```
    <h:panelGrid columns="3">
```

```
        <f:validateBean>
```

```
            <h:outputText value="REF"></h:outputText>
```

```
            <h:inputText id="reference" value="#{catalogueMB.produit.reference}">
```

```
            </h:inputText>
```

```
            <h:message for="reference" errorClass="errors"></h:message>
```

```
            <h:outputText value="DESIGNATION"></h:outputText>
```

```
            <h:inputText id="designation" value="#{catalogueMB.produit.designation}"/>
```

```
            <h:message for="designation" errorClass="errors"></h:message>
```

```
            <h:outputText value="PRIX"></h:outputText>
```

```
            <h:inputText id="prix" value="#{catalogueMB.produit.prix}"/>
```

```
            <h:message for="prix" errorClass="errors"></h:message>
```

```
            <h:outputText value="QUANTITE"></h:outputText>
```

```
            <h:inputText id="quantite" value="#{catalogueMB.produit.quantite}"/>
```

```
            <h:message for="quantite" errorClass="errors"></h:message>
```

```
            <h:commandButton value="Save" action="#{catalogueMB.saveProduit}"/>
```

```
        </f:validateBean>
```

```
    </h:panelGrid>
```

```
</div>
```

REF	<input type="text"/>
DESIGNATION	<input type="text"/>
PRIX	<input type="text" value="0.0"/>
QUANTITE	<input type="text" value="0"/>
<input type="button" value="Save"/>	

Page JSF : index.xhtml

```
<div>
  <h:dataTable value="#{catalogueMB.ListProduits}" var="p" class="table1">
    <h:column>
      <f:facet name="header">
        <h:outputText value="REF"/>
      </f:facet>
      <h:outputText value="#{p.reference}"/>
    </h:column>
    <h:column>
      <f:facet name="header">
        <h:outputText value="DESIGNATION"/>
      </f:facet>
      <h:outputText value="#{p.designation}"/>
    </h:column>
    <h:column>
      <f:facet name="header">
        <h:outputText value="PRIX"/>
      </f:facet>
      <h:outputText value="#{p.prix}"/>
    </h:column>
```

REF	DESIGNATION	PRIX	QUANTITE		
AEA	Imprimante 543	6000.0	10	Supp	Edit
AI321	Smart phone LG	5432.0	14	Supp	Edit
HP654	Ordinateur HP564	8000.0	12	Supp	Edit

Page JSF : index.xhtml

```
<h:column>
    <f:facet name="header">
        <h:outputText value="QUANTITE"></h:outputText>
    </f:facet>
    <h:outputText value="#{p.quantite}"></h:outputText>
</h:column>
<h:column>
    <h:commandLink action="#{catalogueMB.deleteProduit(p.reference)}" value="Supp"/>
</h:column>
<h:column>
    <h:commandLink action="#{catalogueMB.editProduit(p.reference)}" value="Edit"/>
</h:column>
</h:dataTable>
</div>
</h:form>
</body>
</html>
```

REF	DESIGNATION	PRIX	QUANTITE		
AEA	Imprimante 543	6000.0	10	Supp	Edit
AI321	Smart phone LG	5432.0	14	Supp	Edit
HP654	Ordinateur HP564	8000.0	12	Supp	Edit