

De l'esprit à la machine  
L'approche Professo-Académique

**J2EE**

**Abdelahad SATOUR**

# Séances 17–18

JSF

# Concepts techniques clés

- Injection de dépendances
- Programmation Orientée Aspects / Interception
- Annotations
- Mapping Objet / Relationnel
- Sécurité
- Transaction

# Positionnement et architecture JSF

- Contexte création JSF
- Versions et Implémentations
- JSF dans J2EE
- JSF et le pattern MVC
- Cycle de vie requête JSF
- Packages et composants JSF
- Composants additionnels : RichFaces / IceFaces

# Contexte développement d'applications Java/J2ee en 2001

## ➤ **JSP + Servlets**

- Force: bien maîtrisé développeurs
- Faiblesse: maintenance !

## ➤ **Struts**

- Force: Framework web MVC2
- Faiblesse: création interface
- Faiblesse: peu d'outils RAD

## ➤ **Swing :**

- Force: Composants graphiques
- Force: Outils RAD
- Faiblesse: pas d'équivalent web !

## ➤ **Besoin framework web MVC RAD**

# Naissance de JSF

## ➤ **Besoin framework et outils web RAD**

- 'Composants' visuels
- Gestion état écran
- Gestion évènements client
- Multilingue
- Validation entrée utilisateur
- Affichage adapté (pda,portable...)
- Accessibilité
- Navigation entre page

## ➤ **Acteurs** : IBM, SUN, BEA, ORACLE

## ➤ **Mars 2004** : spécifications JSF 1.0

## ➤ **Mai 2004** : spécifications JSF 1.1



Amy Fowler



Craig Mac Clanahan



www.objjis.com -  
INTEGRATION  
CONTINUEwww.objjis.com - 6  
Formation SPRING

# Versions JSF et API J2EE

- **JSF 1.1** (mai 2004)
  - JSR 127 : <http://jcp.org/en/jsr/detail?id=127>
  - servlet 2.3 + JSP 1.2 + Java 1.3
  
- **JSF 1.2** (août 2006)
  - JSR 252 : <http://jcp.org/en/jsr/detail?id=252>
  - Servlet 2.5 + JSP 2.1 + Java 5
  
- **JSF 2.0** (juillet 2009)
  - JSR 314 : <http://jcp.org/en/jsr/detail?id=314>
  - Servlet 2.5 + JSP 2.1 + Java 5+
  - Profil web JEE6

# Implémentations de JSF

- **Mojarra : implémentation de référence (SUN)**

- Bibliothèques : jsf-api.jar, jsf-impl.jar
- Composants natifs : core, html

- **MyFaces : implémentation Apache**

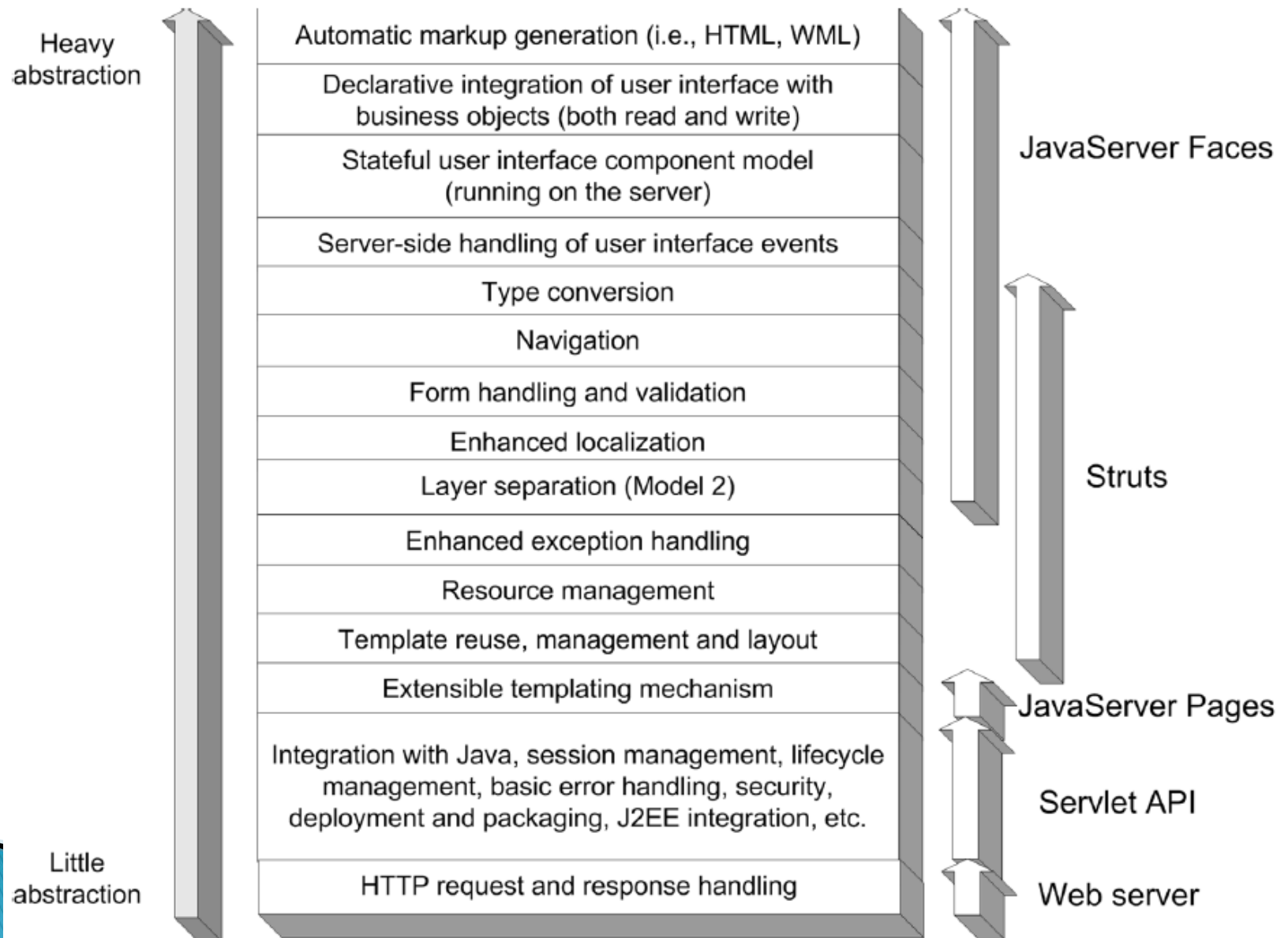
- Bibliothèques : myfaces-api, myfaces-impl
- Composants : TomaHawk, Tobago, Trinidad

- **Nombreuses autres**

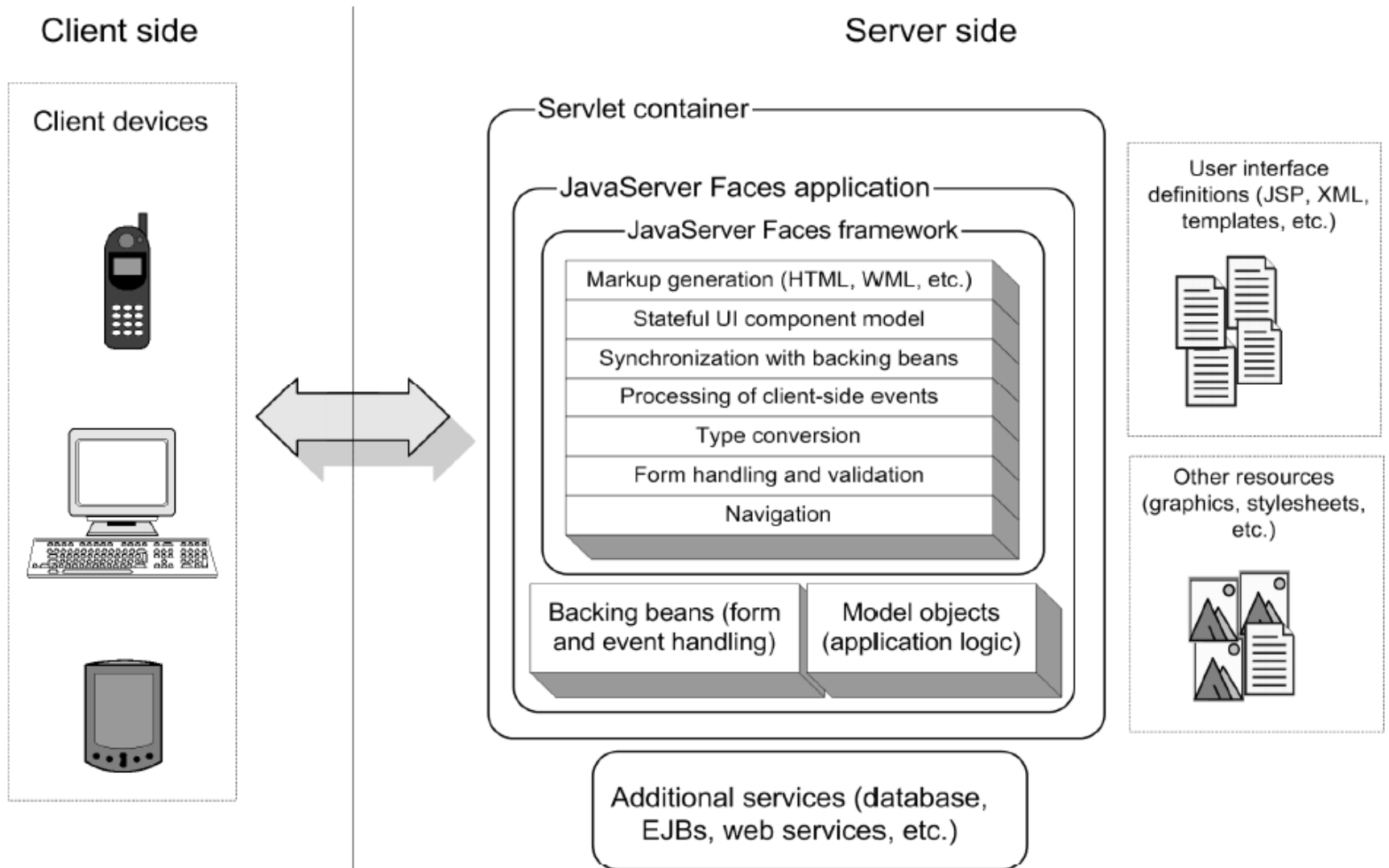
- Détail et comparaison : [www.jsfmatrix.net](http://www.jsfmatrix.net)



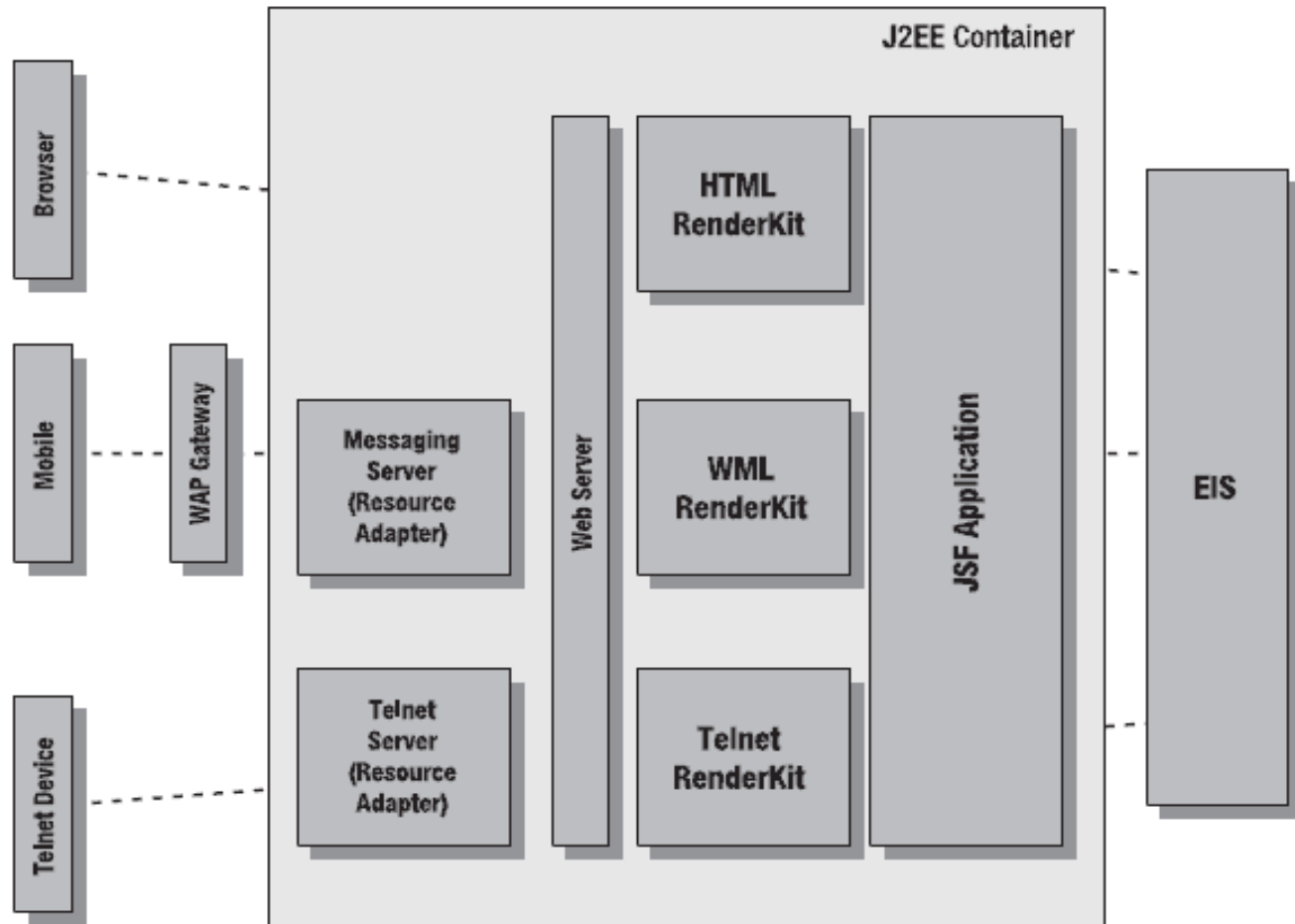
# Architecture & positionnement JSF



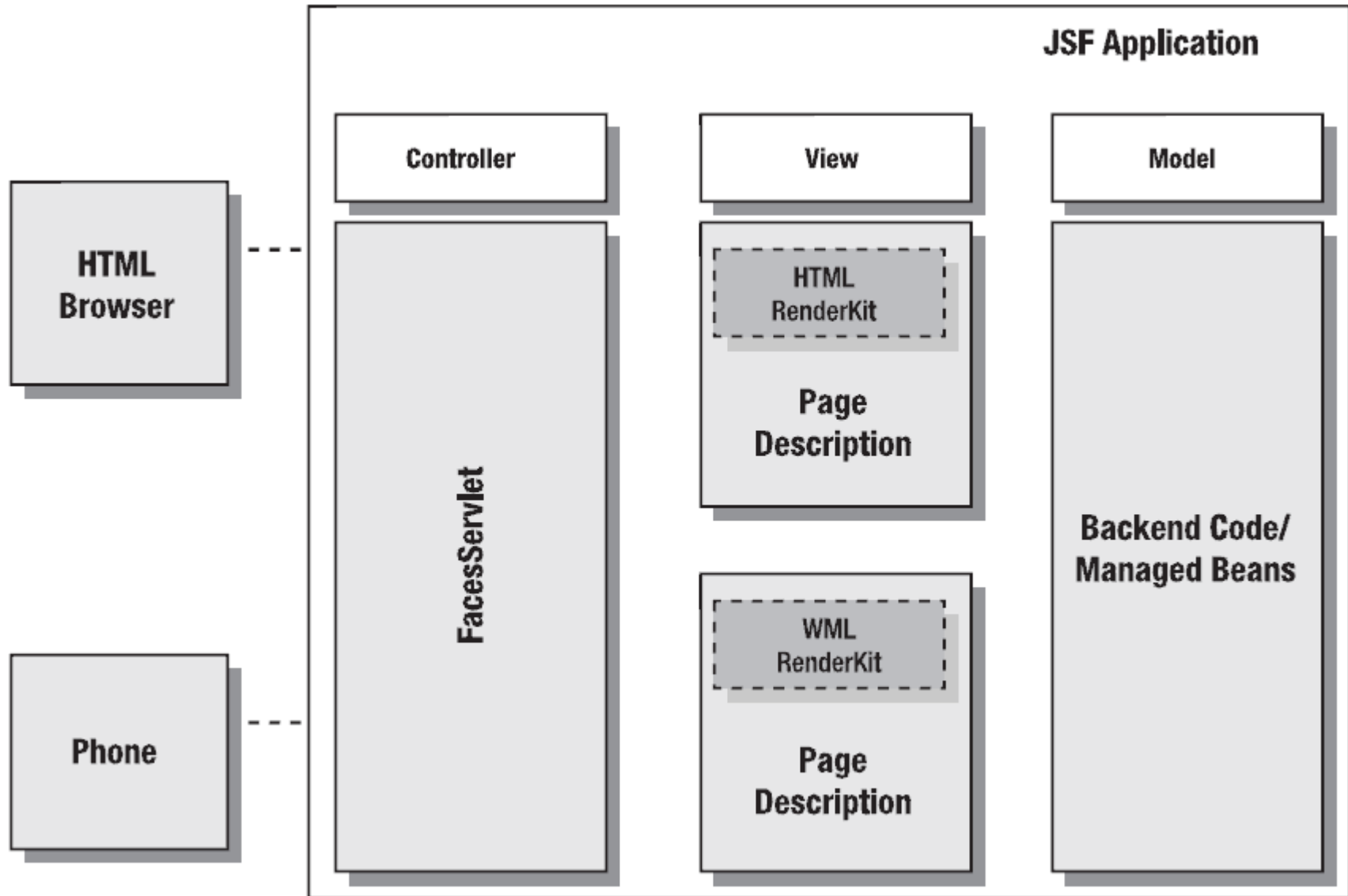
# Architecture & positionnement JSF



# JSF dans application J2EE



# JSF et le modèle MVC

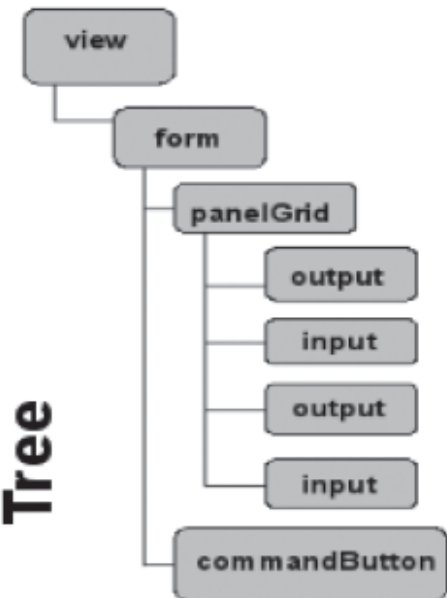


# Exemple

## 1 Development

```
<h:form>
  <h:panelGrid columns="2">
    <h:outputText value="#{msg.emailLabel}"/>
    <h:inputText value="#{login.email}"/>
    <h:outputText value="#{msg.passwordLabel}"/>
    <h:inputText value="#{login.password}"/>
  </h:panelGrid>
  <h:commandButton action="#{login.check}"
    value="#{msg.btnLabel}"/>
</h:form>
```

## 2 JSF Component Tree



## 3 Output

### HTML

```
<td>Email:</td>
<td><input type="text" name="j_id2:j_id5" value="" /></td>
</tr>
<tr>
<td>Password:</td>
<td><input type="text" name="j_id2:j_id7" value="" /></td>
</tr>
</tbody>
</table>
<input type="submit" name="j_id2:j_id8" value="Sign In" />
</form>
```

### Browser

Email:

Password:

# Caractéristiques JSF

- Développement évènementiel orienté Composant

```
<h:commandLink action="register">  
    <h:outputText value="Cliquez ici pour vous enregistrer..." />  
</h:commandLink>
```

---

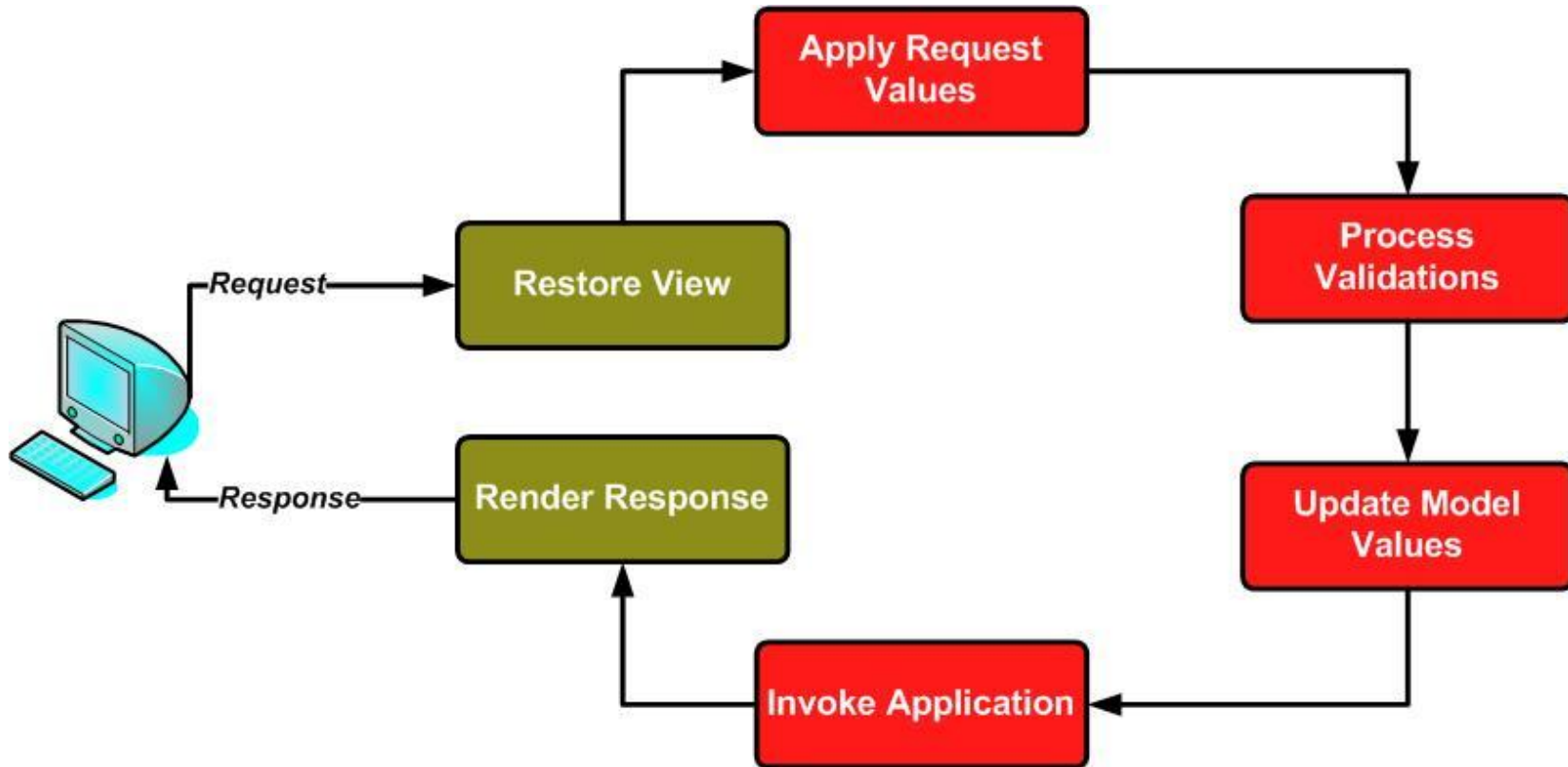
- Simplifie accès et gestion données métiers
- Automatise & simplifie gestion état écran entre plusieurs requêtes
- Mise en oeuvre de patterns connus
- Masque bas niveau (HttpServletRequest/Response)
- Meilleure maintenance

# Modèle évènementiel JSF

- Evènements liés aux 'actions' utilisateurs
  - Appui bouton, clic sur lien hypertexte....
  - Tout composant implémentant 'ActionSource'
- Evènements liés à MAJ interface par l'utilisateur
  - Modification données champ texte, combo
  - Tout composant implémentant 'EditableValueHolder'
- Evènements liés aux données du modèle métier
- Les 'listeners' traitent les évènements

```
<h:commandLink action="register">  
  <h:outputText value="Cliquez ici pour vous enregistrer..." />  
</h:commandLink>
```

# Cycle de requêtes JSF





# **Cycle de vie requête JSF :**

## **6 étapes clés ordonnées**

- **1) Restauration vue (Arbre composants écran)**
- **2) Appliquer paramètres requête**
- **3) Validation (& conversion) entrées utilisateur**
- **4) Mise à jour des objets Modèle (backing beans)**
- **5) Exécution Appli (ActionEvent, ValueChangeEvent...)**
- **6) Rendu de la réponse (Render)**

# Packages / Javadoc JSF

## JSF 1.1

### Packages

- [javax.faces](#)
- [javax.faces.application](#)
- [javax.faces.component](#)
- [javax.faces.component.html](#)
- [javax.faces.context](#)
- [javax.faces.convert](#)
- [javax.faces.el](#)
- [javax.faces.event](#)
- [javax.faces.lifecycle](#)
- [javax.faces.model](#)
- [javax.faces.render](#)
- [javax.faces.validator](#)
- [javax.faces.webapp](#)

## JSF 1.2

### Packages

- [javax.faces](#)
- [javax.faces.application](#)
- [javax.faces.component](#)
- [javax.faces.component.html](#)
- [javax.faces.context](#)
- [javax.faces.convert](#)
- [javax.faces.el](#)
- [javax.faces.event](#)
- [javax.faces.lifecycle](#)
- [javax.faces.model](#)
- [javax.faces.render](#)
- [javax.faces.validator](#)
- [javax.faces.webapp](#)

## JSF 2.0

### Packages

- [javax.faces](#)
- [javax.faces.application](#)
- [javax.faces.component](#)
- [javax.faces.component.behavior](#)
- [javax.faces.component.html](#)
- [javax.faces.component.visit](#)
- [javax.faces.context](#)
- [javax.faces.convert](#)
- [javax.faces.el](#)
- [javax.faces.event](#)
- [javax.faces.lifecycle](#)
- [javax.faces.model](#)
- [javax.faces.render](#)
- [javax.faces.validator](#)
- [javax.faces.view](#)
- [javax.faces.view.facelets](#)
- [javax.faces.webapp](#)

# Composants JSF

Comportement

[javax.faces.component](#)

Interfaces

[ActionSource](#)

[ActionSource2](#)

[ContextCallback](#)

[EditableValueHolder](#)

[NamingContainer](#)

[StateHolder](#)

[ValueHolder](#)

Classes

[UIColumn](#)

[UICommand](#)

[UIComponent](#)

[UIComponentBase](#)

[UIData](#)

[UIForm](#)

[UIGraphic](#)

[UIInput](#)

[UIMessage](#)

[UIMessages](#)

[UINamingContainer](#)

[UIOutput](#)

[UIPanel](#)

[UIParameter](#)

[UISelectBoolean](#)

[UISelectItem](#)

[UISelectItems](#)

[UISelectMany](#)

[UISelectOne](#)

[UIViewRoot](#)

Composant

# Composants JSF rendu HTML disponible avec Mojarra

[javax.faces.component.html](http://javax.faces.component.html)

Classes

[HtmlColumn](#)

[HtmlCommandButton](#)

[HtmlCommandLink](#)

[HtmlDataTable](#)

[HtmlForm](#)

[HtmlGraphicImage](#)

[HtmlInputHidden](#)

[HtmlInputSecret](#)

[HtmlInputText](#)

[HtmlInputTextarea](#)

[HtmlMessage](#)

[HtmlMessages](#)

[HtmlOutputFormat](#)

[HtmlOutputLabel](#)

[HtmlOutputLink](#)

[HtmlOutputText](#)

[HtmlPanelGrid](#)

[HtmlPanelGroup](#)

[HtmlSelectBooleanCheckbox](#)

[HtmlSelectManyCheckbox](#)

[HtmlSelectManyListbox](#)

[HtmlSelectManyMenu](#)

[HtmlSelectOneListbox](#)

[HtmlSelectOneMenu](#)

[HtmlSelectOneRadio](#)

25 composants : pas assez de choix !

[www.objjis.com](http://www.objjis.com) - Formation JSF

[www.objjis.com](http://www.objjis.com) -

INTEGRATION

CONTINUE [www.objjis.com](http://www.objjis.com) - 20

Formation SPRING

# Exemples Composants : Héritage et comportement spécifique

javax.faces.component.html

## Class HtmlCommandButton

java.lang.Object

- └ [javax.faces.component.UIComponent](#)
  - └ [javax.faces.component.UIComponentBase](#)
    - └ [javax.faces.component.UICommand](#)
      - └ javax.faces.component.html.HtmlCommandButton

All Implemented Interfaces:

[ActionSource](#), [ActionSource2](#), [StateHolder](#)

javax.faces.component.html

## Class HtmlInputText

java.lang.Object

- └ [javax.faces.component.UIComponent](#)
  - └ [javax.faces.component.UIComponentBase](#)
    - └ [javax.faces.component.UIOutput](#)
      - └ [javax.faces.component.UIInput](#)
        - └ javax.faces.component.html.HtmlInputText

All Implemented Interfaces:

[EditableValueHolder](#), [StateHolder](#), [ValueHolder](#)

# Packages / Javadoc JSF : Le contexte FacesContext

JSF 1.2

Packages

[javax.faces](#)  
[javax.faces.application](#)  
[javax.faces.component](#)  
[javax.faces.component.html](#)  
[javax.faces.context](#)  
[javax.faces.convert](#)  
[javax.faces.el](#)  
[javax.faces.event](#)  
[javax.faces.lifecycle](#)  
[javax.faces.model](#)  
[javax.faces.render](#)  
[javax.faces.validator](#)  
[javax.faces.webapp](#)

[javax.faces.context](#)

Classes

[ExternalContext](#)  
[FacesContext](#)  
[FacesContextFactory](#)  
[ResponseStream](#)  
[ResponseWriter](#)  
[ResponseWriterWrapper](#)

## Method Summary

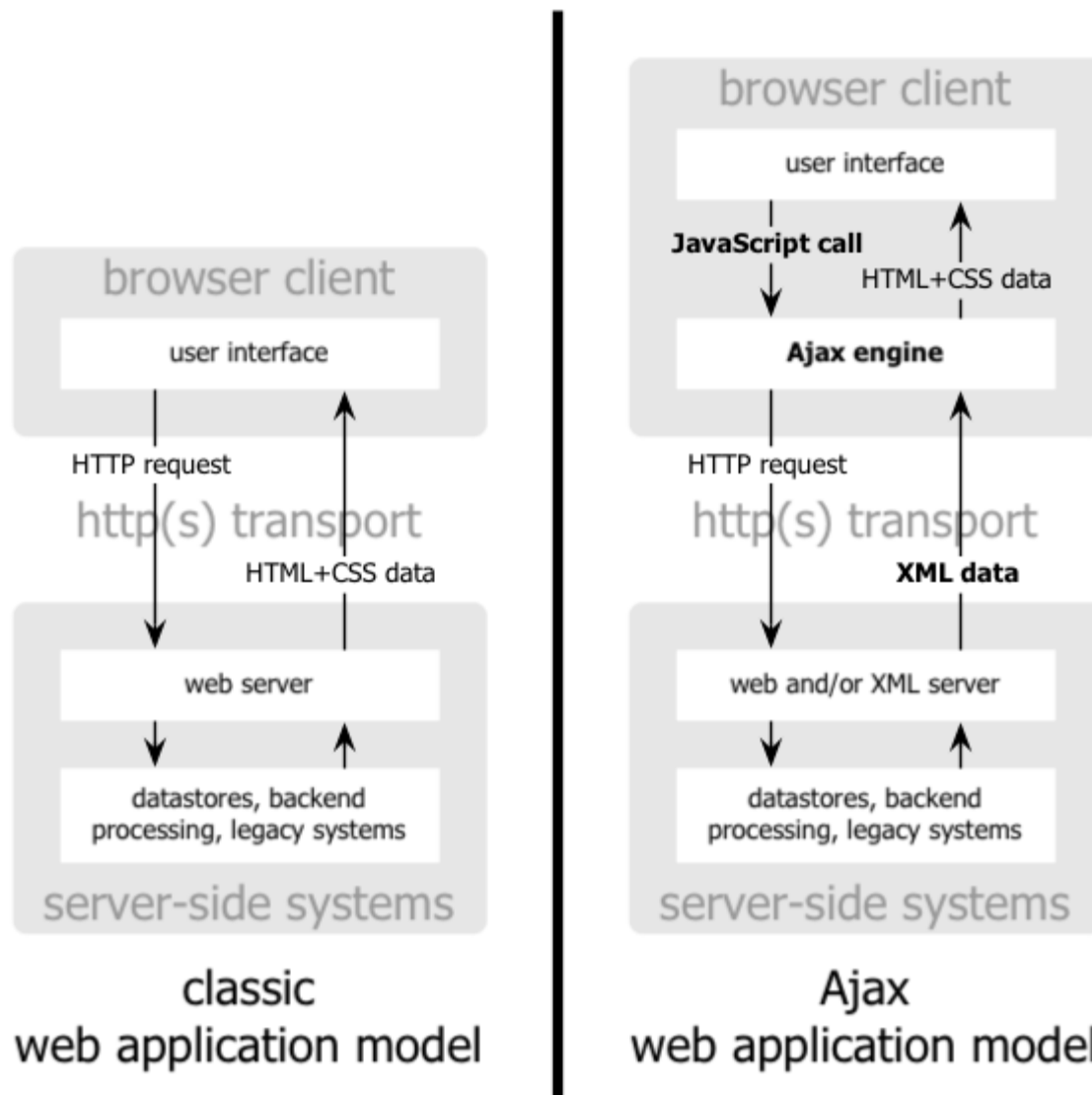
abstract void	<a href="#">addMessage</a> (java.lang.String clientId, <a href="#">FacesMessage</a> message) Append a <a href="#">FacesMessage</a> to the set of messages associated with the specified client identifier, if clientId is not null.
abstract <a href="#">Application</a>	<a href="#">getApplication</a> () Return the <a href="#">Application</a> instance associated with this web application.
abstract <a href="#">java.util.Iterator</a> < <a href="#">java.lang.String</a> >	<a href="#">getClientIdsWithMessages</a> () Return an Iterator over the client identifiers for which at least one <a href="#">FacesMessage</a> has been queued.
static <a href="#">FacesContext</a>	<a href="#">getCurrentInstance</a> () Return the <a href="#">FacesContext</a> instance for the request that is being processed by the current thread, if any.
<a href="#">javax.el.ELContext</a>	<a href="#">getELContext</a> () Return the <a href="#">ELContext</a> instance for this <a href="#">FacesContext</a> instance.
abstract <a href="#">ExternalContext</a>	<a href="#">getExternalContext</a> () Return the <a href="#">ExternalContext</a> instance for this <a href="#">FacesContext</a> instance.
abstract <a href="#">FacesMessage.Severity</a>	<a href="#">getMaximumSeverity</a> () Return the maximum severity level recorded on any <a href="#">FacesMessages</a> that has been queued, whether or not they are associated with any specific <a href="#">UIComponent</a> .
abstract <a href="#">java.util.Iterator</a> < <a href="#">FacesMessage</a> >	<a href="#">getMessages</a> () Return an Iterator over the <a href="#">FacesMessages</a> that have been queued, whether or not they are associated with any specific client identifier.
abstract <a href="#">java.util.Iterator</a> < <a href="#">FacesMessage</a> >	<a href="#">getMessages</a> (java.lang.String clientId) Return an Iterator over the <a href="#">FacesMessages</a> that have been queued that are associated with the specified client identifier (clientId is not null), or over the <a href="#">FacesMessages</a> that have been queued that are not associated with any specific client identifier (clientId is null).
abstract <a href="#">RenderKit</a>	<a href="#">getRenderKit</a> () Return the <a href="#">RenderKit</a> instance for the render kit identifier specified on our <a href="#">UIViewRoot</a> , if there is one.

# Composants JSF supplémentaires

**RichFaces (Jboss) : [www.jboss.org/richfaces](http://www.jboss.org/richfaces)**

- **Fusion projets Jboss 'RichFaces' et 'Ajax4jsf'**
  - **Le développeur choisit les zones de la page à traiter coté serveur + zones à mettre à jour**
  - **+ de 100 composants UI avec support Ajax testés avec différents navigateurs**
- **Chaque composant : + de 30 attributs**
- **Tags (a4j: et rich:)**
- **Skins (DEFAULT, blueSky, plain, laguna...)**
- **CDK (Components Development Kit)**
- **IceFaces ([www.icefaces.org](http://www.icefaces.org))**
- **Composants MyFaces (<http://myfaces.apache.org>)**

# Rappels Ajax

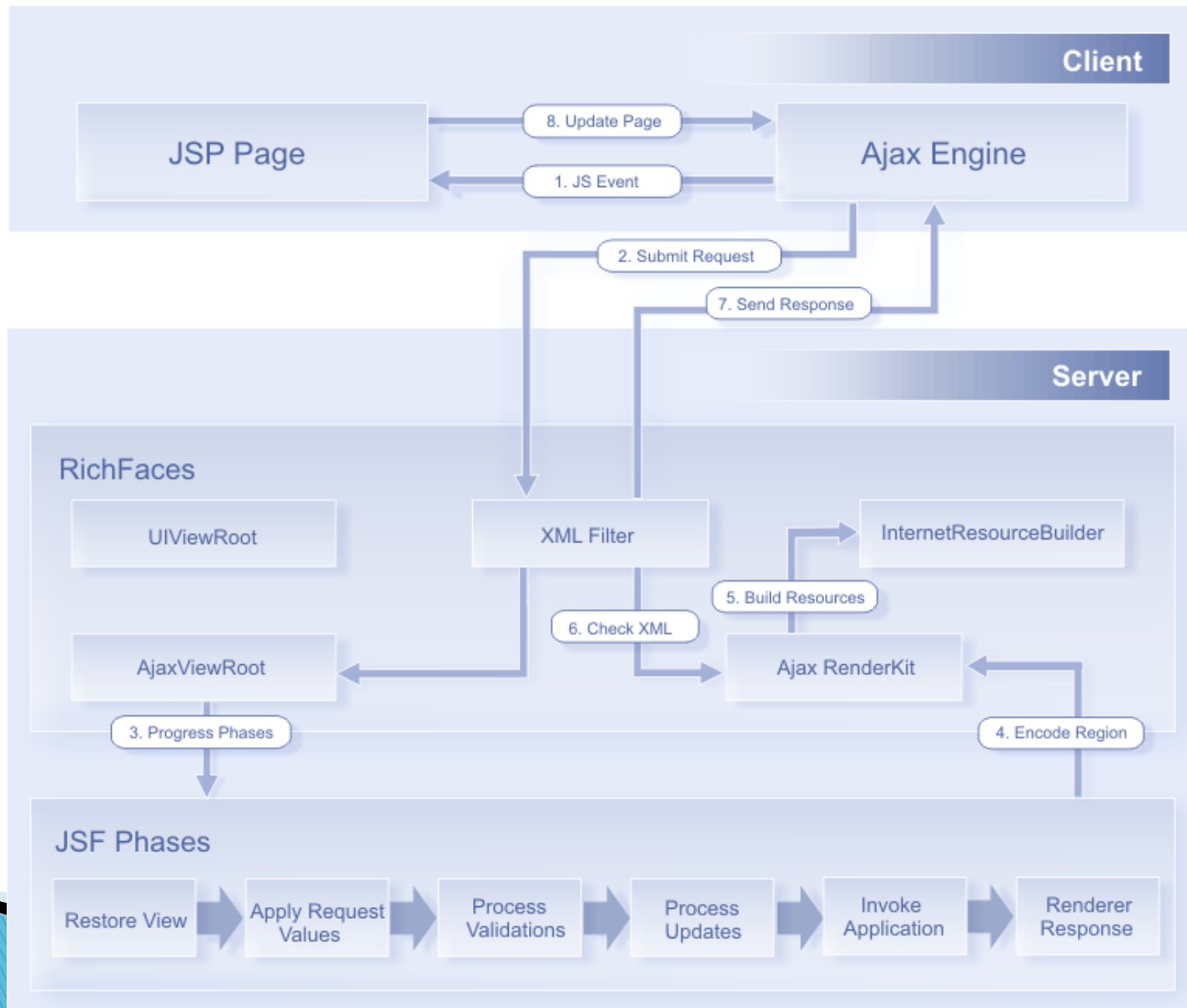




# JSF et Ajax avec RichFaces

- **Ajax4jsf** : né de la volonté de créer applications riches / Ajax avec JSF
- **Ajax4jsf** : filtre qui ajoute des fonctions javascript ainsi que les dépendances de l'objet XMLHttpRequest aux composants JSF
- Suite à requête HTTP, le moteur Ajax transforme la requête et l'envoi au filtre Ajax4jsf
- Le filtre Ajax4jsf convertit les données au format XML et transfère la requête à la servlet FacesServlet, pour réaliser cycle 6 étapes JSF

# JSF et Ajax avec RichFaces

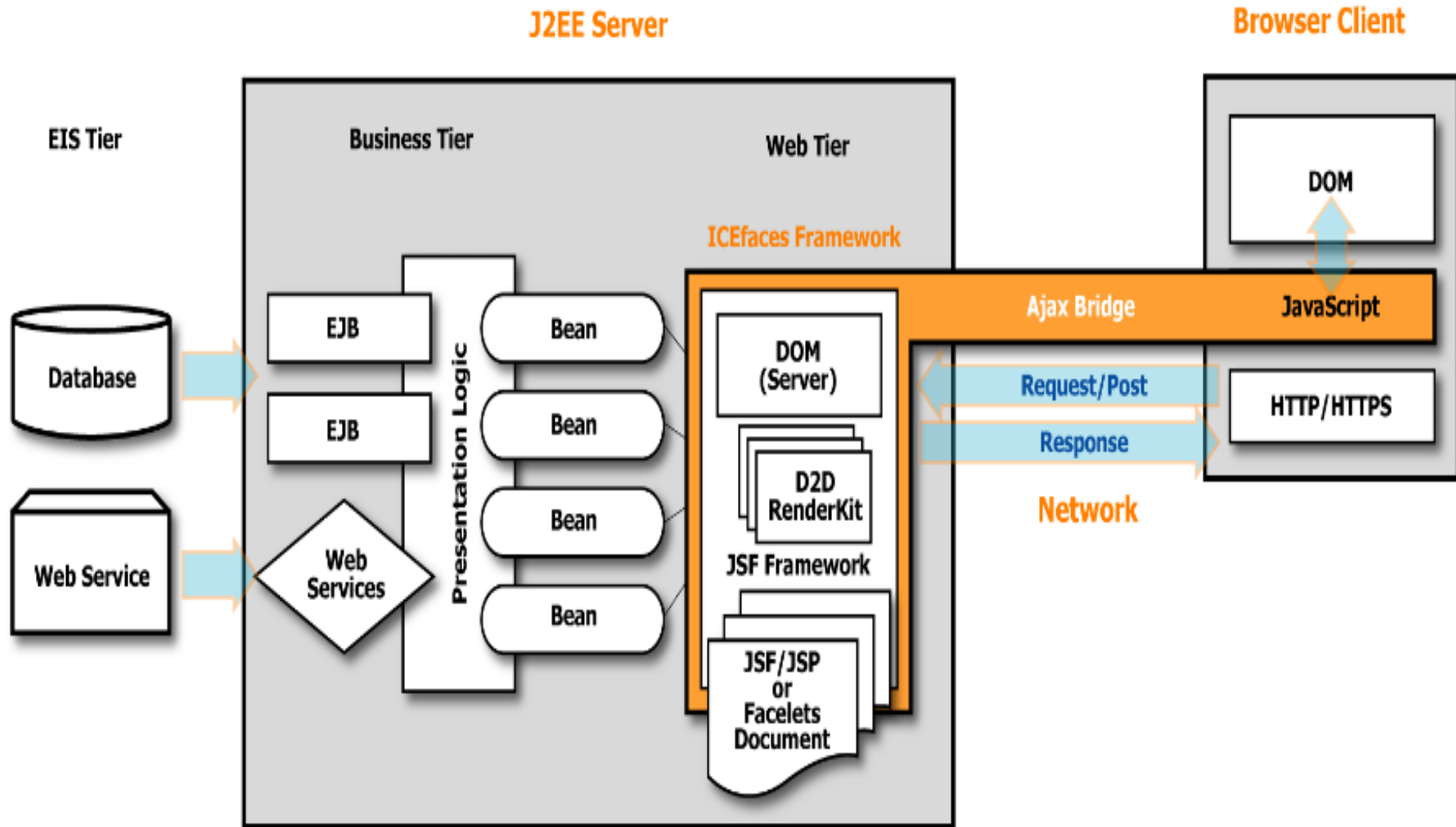


# Composants JSF supplémentaires

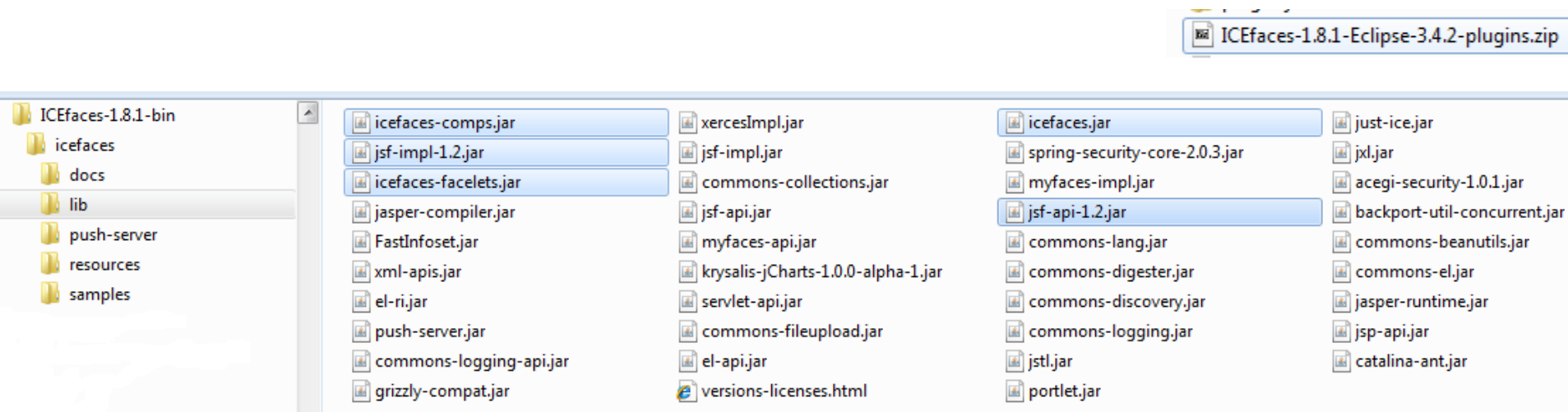
**IceFaces (IceSoft) :** [www.icefaces.org](http://www.icefaces.org)

- **Leader historique sur JSF + Ajax/Web2**
- **Technologies Skinning, Multimedia, AJAX Push**
- **Intégration avec**
  - **JSF 1.x / JSF 1.2 / Facelet**
  - **Spring Web Flow**
  - **Jboss Seam**
  - **MyFaces Tomahawk**
  - **Portails : Liferay, Jboss Portal, WebLo. Portal**
  - **IDE : Eclipse, MyEclipse, Netbeans**
- **Communauté : 100.000 développeurs+**

# JSF et Ajax avec IceFaces



# Librairies Icefaces



## Dépendences icefaces 181

Artifact	Version
<a href="#">backport-util-concurrent</a>	2.2
<a href="#">FastInfoset</a>	1.2.2
<a href="#">commons-beanutils</a>	1.8.0
<a href="#">commons-collections</a>	3.2
<a href="#">commons-digester</a>	1.8
<a href="#">commons-fileupload</a>	1.2.1
<a href="#">commons-logging</a>	1.1
<a href="#">commons-logging-api</a>	1.1
<a href="#">el-api</a>	1.0

[www.objjis.com](http://www.objjis.com) –  
 INTEGRATION  
 CONTINUE[www.objjis.com](http://www.objjis.com) – 29  
 Formation SPRING

# IceFaces : plugin Eclipse

- **Intégration du plugin dans Eclipse**
  - **Manuelle si version plugin < 3.6.2**
    - **Exemple : eclipse 3.4 + Icefaces 1.8.1**
  - **Sinon Update site : [icefaces.org/eclipse-updates](http://icefaces.org/eclipse-updates)**
    - **Exemple : dès Eclipse 3.5**
- **Valeur ajoutée du plugin pour développeur :**
  - **Création projets Web JSF ICeFaces**
  - **Création projets Web JSF IceFaces + Facelet**
  - **Complétion de code**
  - **Gestion visuelle composants graphiques**

# IceFaces : Plugin Eclipse

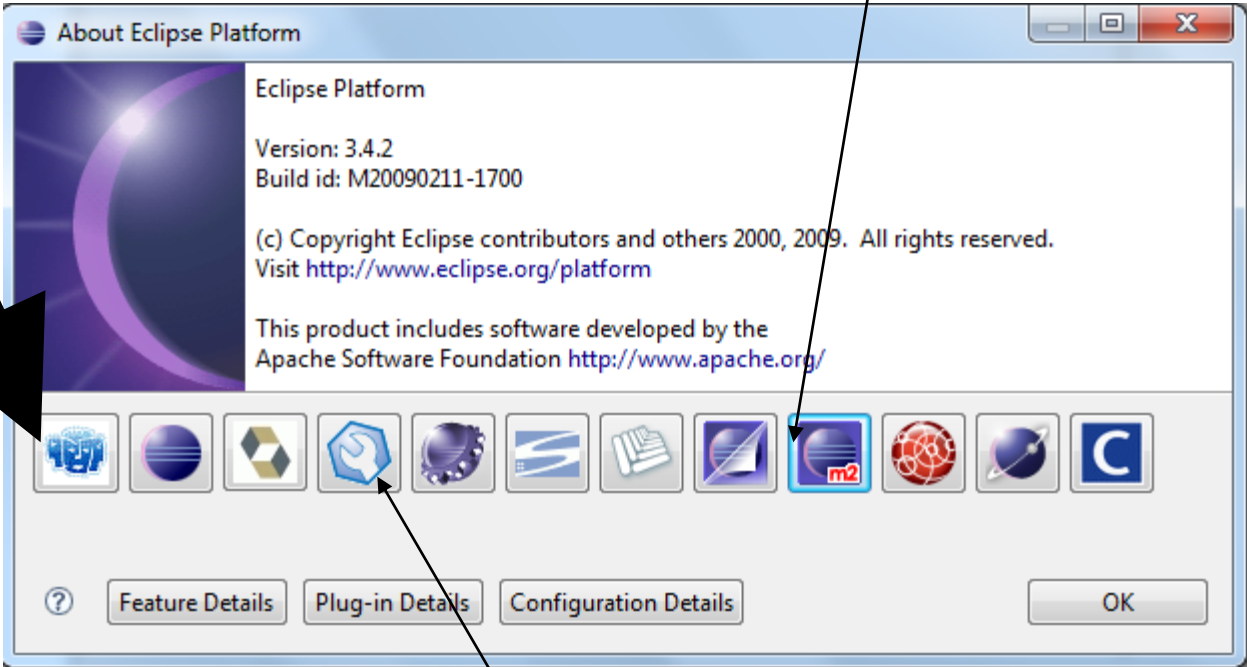
ICEfaces-1.8.1-Eclipse-3.4.2-plugins.zip

Plugin IceFaces (dézipper archive + install eclipse 'local')

Plugin m2eclipse (Maven)

Plugin JbossTools pour intégration Jboss (dézipper le zip dans répertoire 'eclipse')

JBossTools-ALL-win32-3.0.0.GA-R200903141626-H5.zip



# IceFaces : industrialiser développements

## ➤ AppFuse : squelettes projets (archetypes maven)

- [www.appfuse.org](http://www.appfuse.org)
- Initié en 2002 par Mark Raible
- Squelettes appli. Struts/Jsf + hibernate/spring...
- Aucun squelette avec Icefaces en frontal !

## ➤ Edoras :

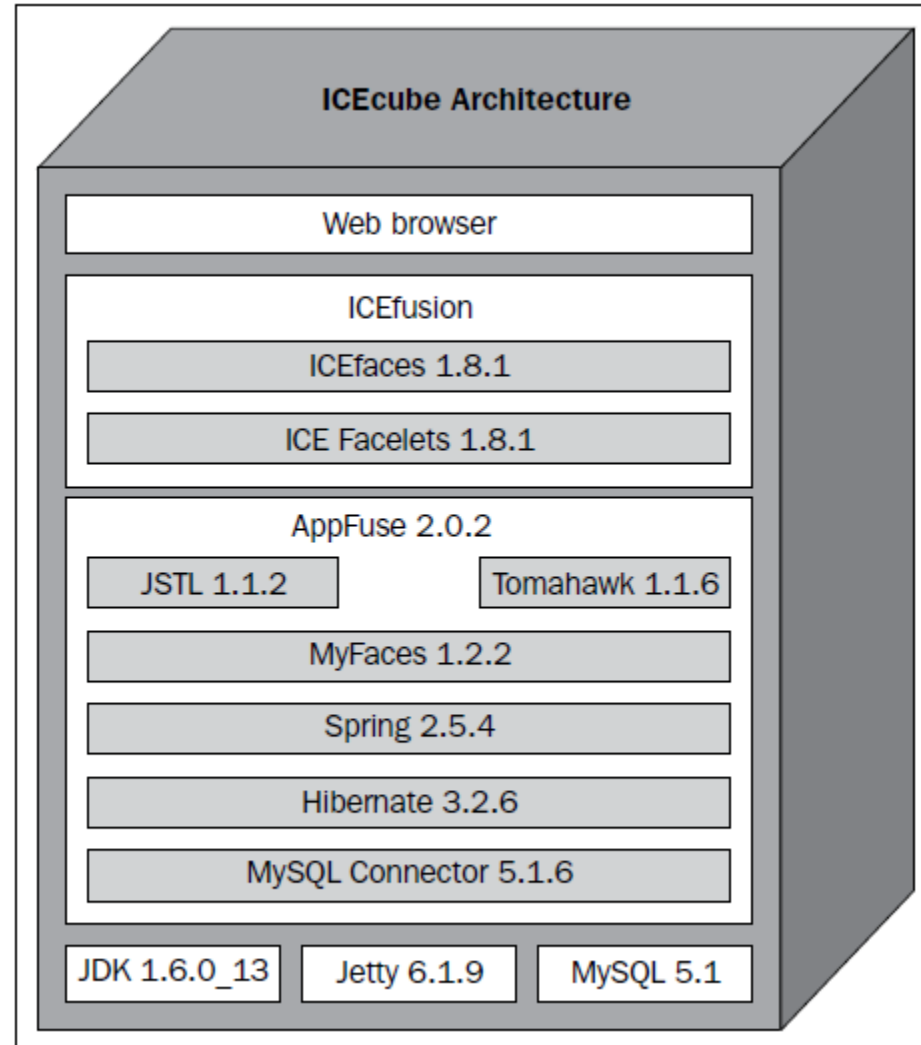
- [www.edorasframework.org](http://www.edorasframework.org)
- Initié par MIMACOM, qui fournit support IceFaces en Europe (mimacom.ch)

IceFusion = AppFuse pour IceFaces

- [icefusion.googlecode.com](http://icefusion.googlecode.com)

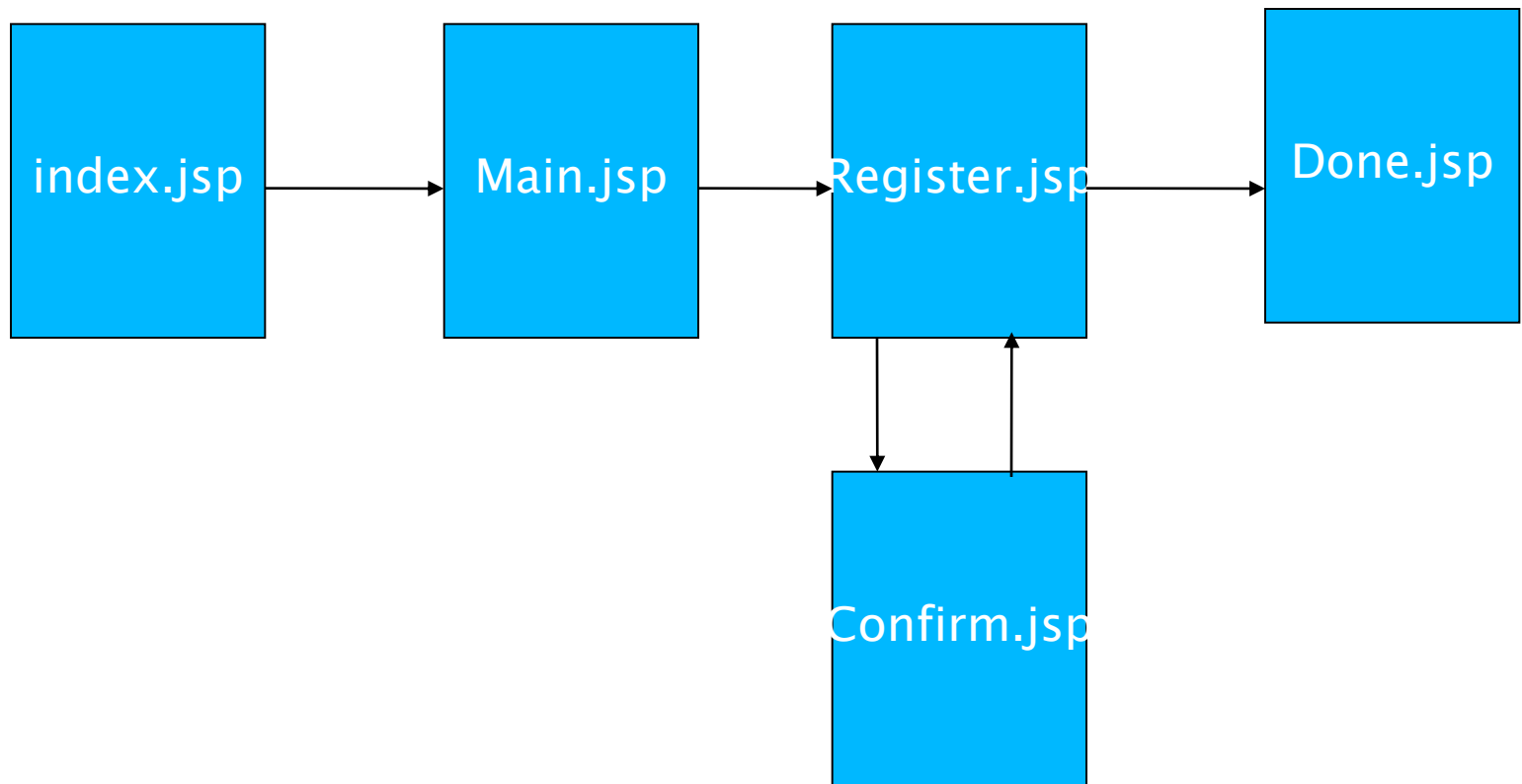


# Développement IceFaces



# Démo JSF :

## Cinématique application



# Configuration JSF : fichier web.xml

web.xml

```
1<?xml version = '1.0' encoding = 'windows-1252'?>
2<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3    version="2.4"    xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
4    xmlns="http://java.sun.com/xml/ns/j2ee">
5
6    <!-- Controleur : la servlet Faces Servlet fournies par JSF -->
7    <servlet>
8        <servlet-name>Faces Servlet</servlet-name>
9        <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
10        <load-on-startup>1</load-on-startup>
11    </servlet>
12
13    <!-- Le controleur intervient pour toutes les URL débutant par /faces/ -->
14    <servlet-mapping>
15        <servlet-name>Faces Servlet</servlet-name>
16        <url-pattern>/faces/*</url-pattern>
17    </servlet-mapping>
18
19    <!-- Page d'accueil de l'application -->
20    <welcome-file-list>
21        <welcome-file>index.html</welcome-file>
22    </welcome-file-list>
23
24</web-app>
25
```

1

2

# Enchaînement accueil → écran JSF

```
index.jsp
1<%@ page contentType="text/html"%>
2<html>
3<head>
4<title>Un exemple d'application JSF</title>
5</head>
6<body>
7<jsp:forward page="/faces/main.jsp" />
8</body>
9</html>
10
```

0

```
main.jsp
1<%@ page contentType="text/html"%>
2
3<%@ taglib uri="http://java.sun.com/jsf/core" prefix="f"%>
4<%@ taglib uri="http://java.sun.com/jsf/html" prefix="h"%>
5
6<f:view>
7<html>
8<head>
9<title>Un formulaire d'enregistrement JSF</title>
10</head>
11<body>
12<h:form>
13<h2>Formulaire d'enregistrement</h2>
14
15<h:commandLink action="register">
16<h:outputText value="Cliquez ici pour vous enregistrer...">
17</h:commandLink>
18</h:form>
19
20<hr>
21
22</body>
23</html>
24</f:view>
```

2

3

1

4

5

# Configuration JSF : faces-config.xml

The diagram shows a code editor with the file 'faces-config.xml' open. Three blue circles with white numbers are used as annotations: Circle 1 points to the root element '<faces-config>', Circle 2 points to the 'xmlns' attribute of the root element, and Circle 3 points to the first '<navigation-rule>' element. The code is as follows:

```
1<?xml version="1.0" encoding="windows-1252"?>
2<!DOCTYPE faces-config PUBLIC "-//Sun Microsystems, Inc.//DTD JavaServer Faces Config 1.1//EN"
3  "http://java.sun.com/dtd/web-facesconfig_1_1.dtd">
4<faces-config xmlns="http://java.sun.com/JSF/Configuration">
5  <managed-bean>
6    <managed-bean-name>UserBean</managed-bean-name>
7    <managed-bean-class>com.objis.demojsf.domaine.UserBean</managed-bean-class>
8    <managed-bean-scope>session</managed-bean-scope>
9  </managed-bean>
10 <navigation-rule>
11   <from-view-id>/main.jsp</from-view-id>
12   <navigation-case>
13     <from-outcome>register</from-outcome>
14     <to-view-id>/register.jsp</to-view-id>
15   </navigation-case>
16 </navigation-rule>
17 <navigation-rule>
18   <from-view-id>/register.jsp</from-view-id>
19   <navigation-case>
20     <from-outcome>register</from-outcome>
21     <to-view-id>/confirm.jsp</to-view-id>
22   </navigation-case>
23 </navigation-rule>
24 <navigation-rule>
25   <from-view-id>/confirm.jsp</from-view-id>
26   <navigation-case>
27     <from-outcome>success</from-outcome>
28     <to-view-id>/done.jsp</to-view-id>
29   </navigation-case>
30   <navigation-case>
31     <from-outcome>revise</from-outcome>
32     <to-view-id>/register.jsp</to-view-id>
33   </navigation-case>
34 </navigation-rule>
35</faces-config>
```

# Intégration JSP / JSF

## ➤ Objectifs différents

- JSP : html + java / traitement 'de haut en bas'
- JSF : composants / traitement évènementiel

## ➤ Cycle de vie différent

- JSP → Java → html
- JSF : cycle de requête précis (6 étapes)

## ➤ Etapes gestion composant JSF

- création + gestion entrée + rendu

## ➤ Etapes composant JSF dans JSP

- En parallèle : création & rendu

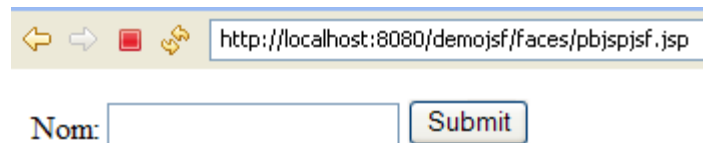
## ➤ Ajout de contenu dans la page par JSF & JSP

- En parallèle : création & rendu !

# Intégration JSP / JSF :

## Exemple de problème

```
5<html>
6<body>
7<f:view>
8  <h:form>
9
10<!-- Reference à un composant cité plus bas :
11      on ne voit pas ce label  au premier affichage-->
12
13<h:outputLabel for="name">
14  <h:outputText value="Nom:" />
15</h:outputLabel>
16
17  <h:inputText id="name" />
18
19  <h:commandButton value="Submit" />
20
21</h:form>
22</f:view>
23</body>
24</html>
```



# Intégration JSP / JSF :

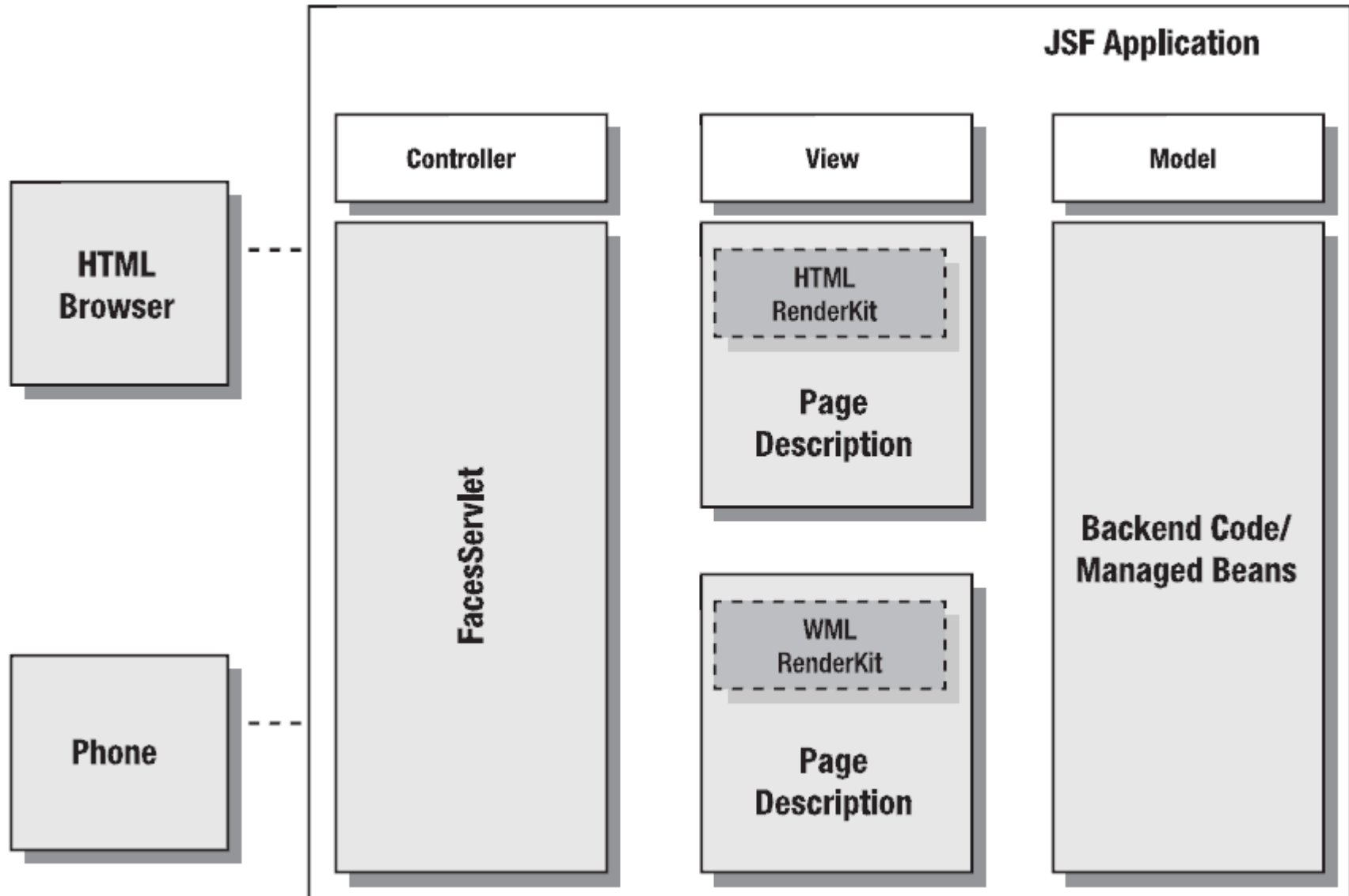
## Comprendre la technologie Facelet

- **Technologie de Vue adaptée à JSF**
  - ViewHandler
- **S'insère bien dans le cycle requête JSF**
- **<http://facelets.dev.java.net>**

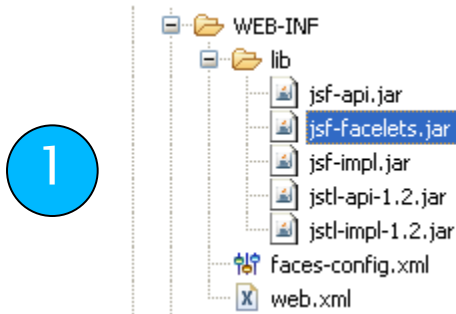


# Intégration JSP / JSF :

## Comprendre la technologie Facelet



# Facelet : Configuration



```
5<faces-config>
6
7  <!-- Specifier le ViewHandler -->
8  <application>
9    <view-handler>
10      com.sun.facelets.FaceletViewHandler
11    </view-handler>
12  </application>
```

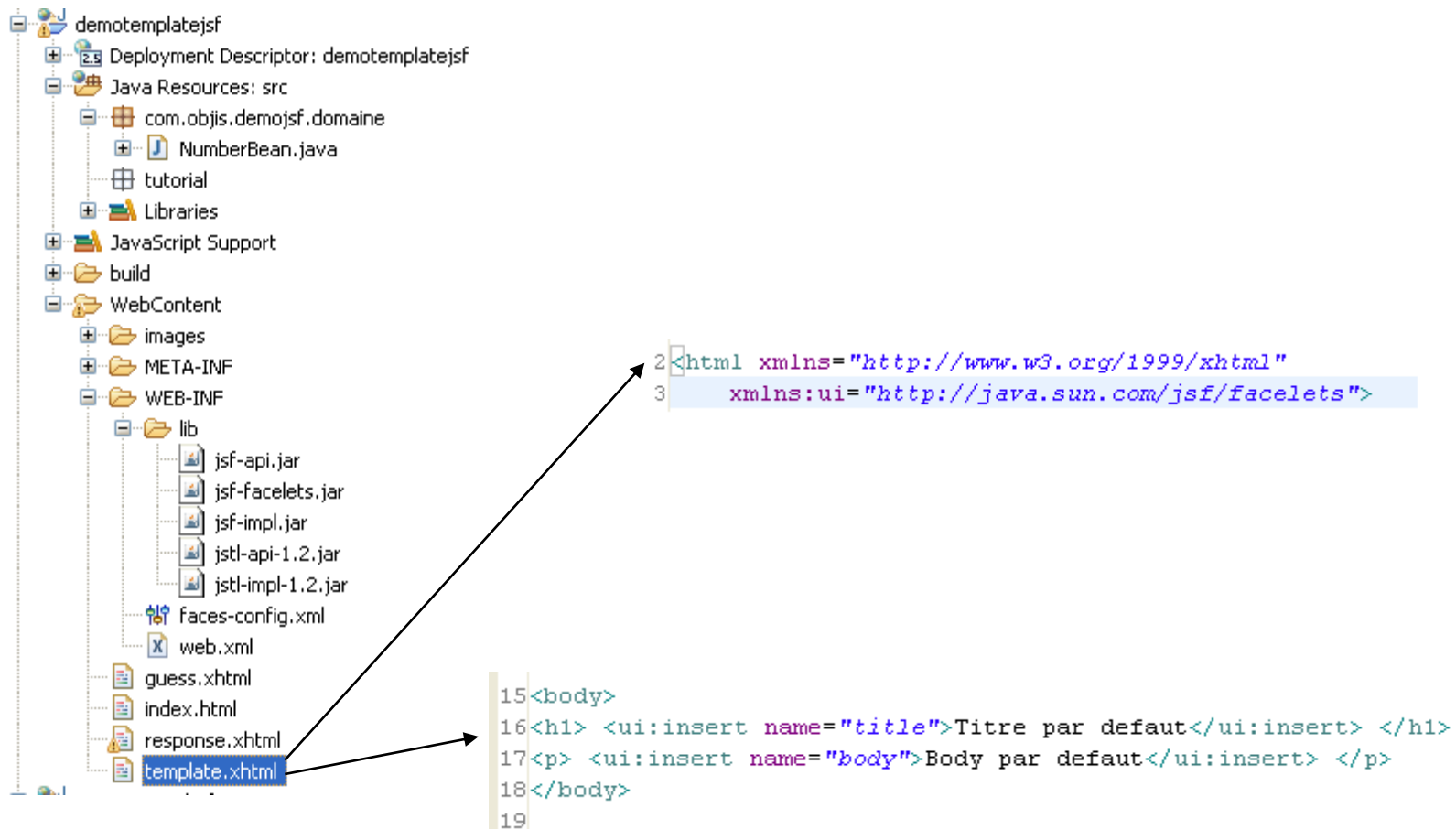
aces-config.xml

3

```
42  <context-param>
43    <param-name>javax.faces.DEFAULT_SUFFIX</param-name>
44    <param-value>.xhtml</param-value>
45  </context-param>
46
```

Web.xml

# Facelet : template.xhtml



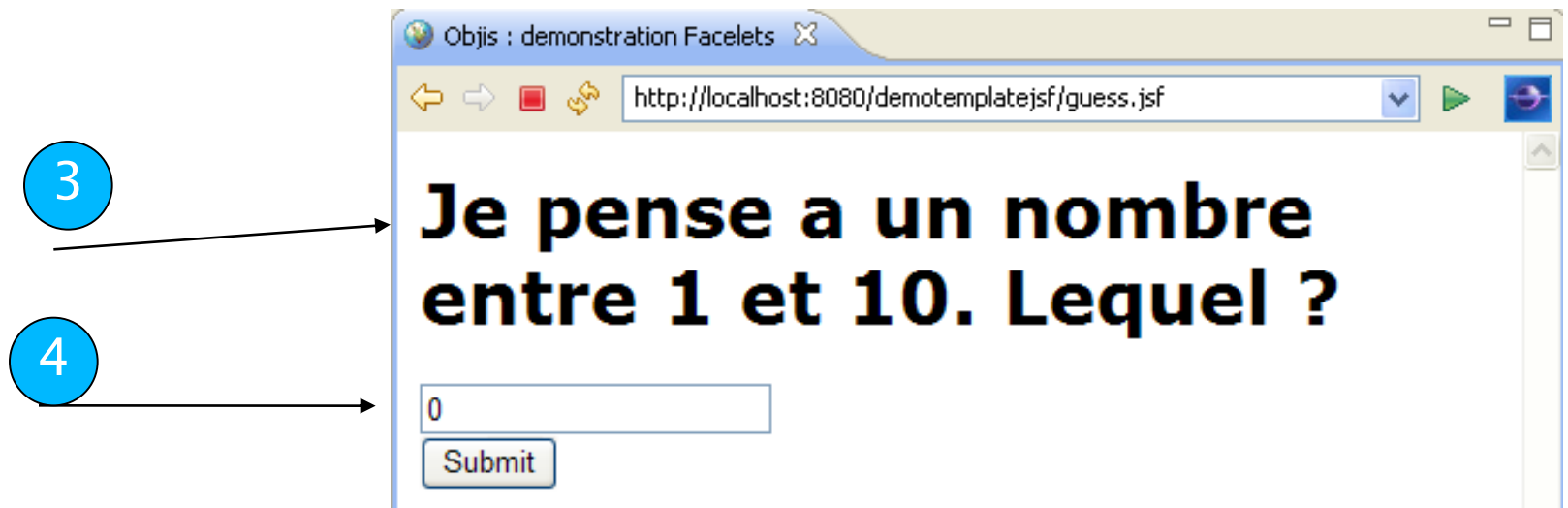
# Facelet :

## Exemple ecran 1 (guess.xhtml)

```
guess.xhtml X
1<!--
2Demonstration Template Facelet JSF : cet ecran est cree a partir du modele template.xhtml
3Il y a 2 zones a redefinir : le titre ("title") de la page et le corps de la page ("body")
4-->
5<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
6<html xmlns="http://www.w3.org/1999/xhtml"
7      xmlns:ui="http://java.sun.com/jsf/facelets" 1
8      xmlns:h="http://java.sun.com/jsf/html">
9<body>
10
11<ui:composition template="/template.xhtml"> 2
12
13  <ui:define name="title">
14    Je pense a un nombre entre #{NumberBean.min} et #{NumberBean.max}. Lequel ?
15  </ui:define>
16
17  <ui:define name="body">
18    <h:form id="helloForm">
19      <h:inputText type="text" id="userNo" value="#{NumberBean.guess}" validator="#{NumberBean.validate}"/>
20      <br/>
21      <h:commandButton type="submit" id="submit" action="success" value="Submit" />
22      <br/>
23      <h:message showSummary="true" showDetail="false" style="color: red; font-weight: bold;" id="errors1" for="userNo"/>
24    </h:form>
25  </ui:define>
26
27</ui:composition>
28
29</body>
30</html>
```

# Facelet :

## rendu ecran 1 (guess.xhtml)



# Facelet :

## Exemple ecran 2 (response.xhtml)

response.xhtml

```
1<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"
2<html xmlns="http://www.w3.org/1999/xhtml"
3      xmlns:ui="http://java.sun.com/jsf/facelets"
4      xmlns:h="http://java.sun.com/jsf/html">
5<body>
6
7<ui:composition template="/template.xhtml">
8
9  <ui:define name="title">
10    #{NumberBean.message}
11  </ui:define>
12
13  <ui:define name="body">
14    <form jsfc="h:form">
15      <input jsfc="h:commandButton" type="submit" id="back" value="Back" action="success"/>
16    </form>
17  </ui:define>
18
19</ui:composition>
20
21</body>
22</html>
```

1

2

3

4

Permet collaboration développeur / webdesigner

# Facelet :

## rendu ecran 2 (response.xhtml)



# Démo Facelet avec IceFaces : Template v1 (merci Eclipse !)

```
page-template.xhtml X
<!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:ui="http://java.sun.com/jsf/facelets"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:ice="http://www.icesoft.com/icefaces/component">
<head>
  <title><ui:insert name="title">Default title</ui:insert></title>
</head>
<body>
  <div id="header">
    <ui:include src="/header.xhtml">
      <ui:param name="param_name" value="param_value"/>
    </ui:include>
  </div>
  <div id="content">
    <ice:form>

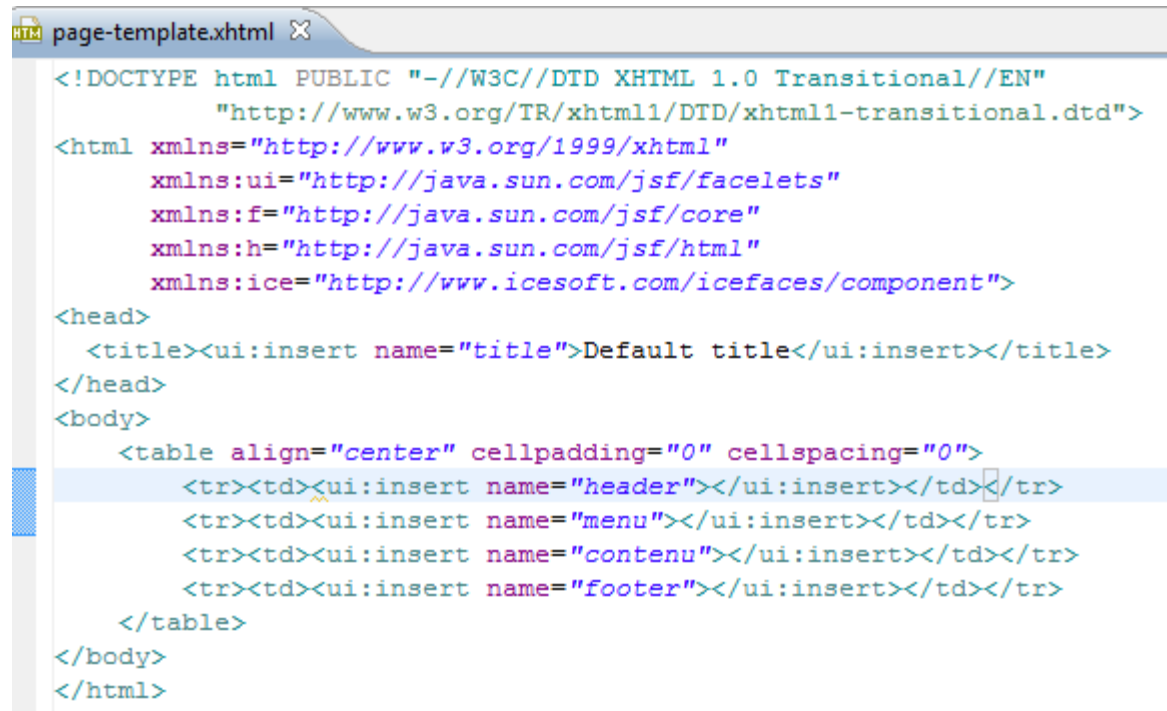
    </ice:form>
  </div>
</body>
</html>
```



# Démo Facelet avec IceFaces : Template v2

```
*page-template.xhtml ✕
<!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:ui="http://java.sun.com/jsf/facelets"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:ice="http://www.icesoft.com/icefaces/component">
<head>
  <title><ui:insert name="title">Default title</ui:insert></title>
</head>
<body>
  <table align="center" cellpadding="0" cellspacing="0">
    <tr><td><!-- Header --></td></tr>
    <tr><td><!-- Menu --></td></tr>
    <tr><td><!-- Contenu --></td></tr>
    <tr><td><!-- Footer --></td></tr>
  </table>
</body>
</html>
```

# Démo Facelet avec IceFaces : Template v3



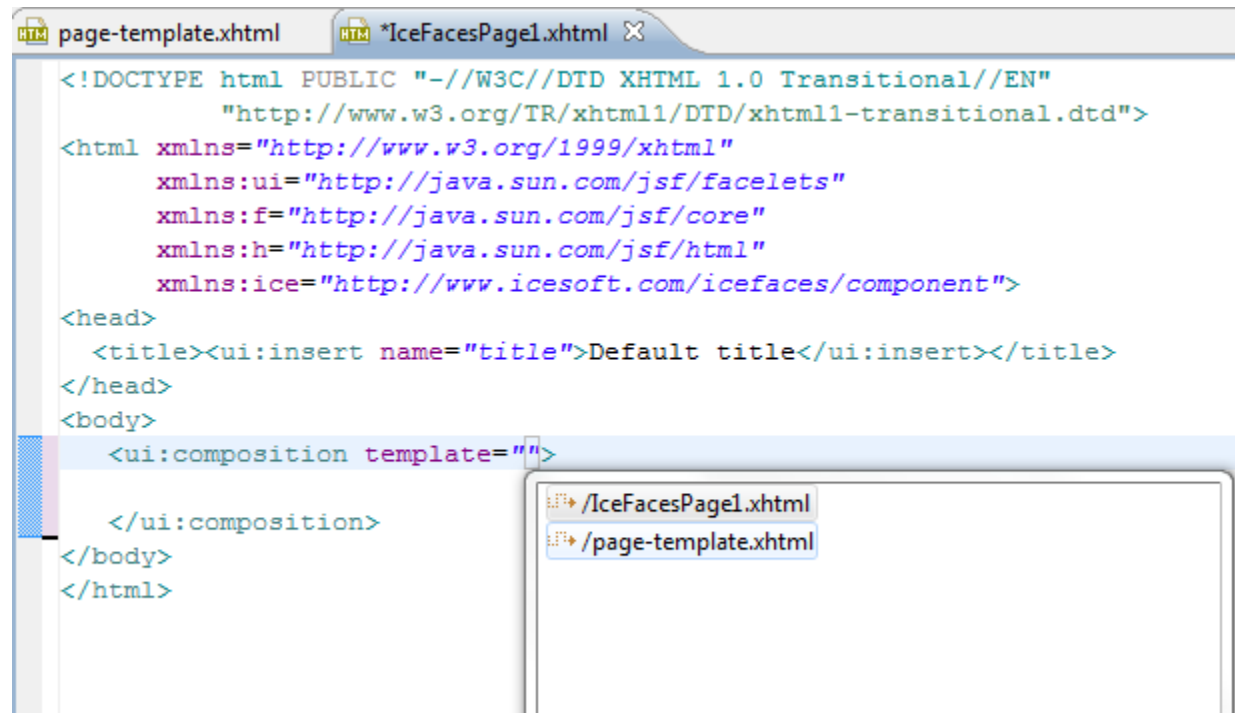
The screenshot shows a web browser window with the title 'page-template.xhtml'. The browser's address bar and the page content display the XML source code of the Facelet template. The code is as follows:

```
<!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:ui="http://java.sun.com/jsf/facelets"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:ice="http://www.icesoft.com/icefaces/component">
<head>
  <title><ui:insert name="title">Default title</ui:insert></title>
</head>
<body>
  <table align="center" cellpadding="0" cellspacing="0">
    <tr><td><ui:insert name="header"></ui:insert></td></tr>
    <tr><td><ui:insert name="menu"></ui:insert></td></tr>
    <tr><td><ui:insert name="contenu"></ui:insert></td></tr>
    <tr><td><ui:insert name="footer"></ui:insert></td></tr>
  </table>
</body>
</html>
```

# Démo Facelet avec IceFaces : Template v4

```
page-template.xhtml
<head>
<ice:outputStyle href="/xmlhttp/css/royale/royale.css"></ice:outputStyle>
<title><ui:insert name="title">TITRE</ui:insert></title>
</head>
<body>
<table align="center" cellpadding="0" cellspacing="0">
  <tr>
    <td><ui:insert name="header">
      <ice:graphicImage url="/images/baniere.png"></ice:graphicImage>
    </ui:insert></td>
  </tr>
  <tr>
    <td><ui:insert name="menu">
      <ice:form>
        <ice:menuBar noIcons="true">
          <ice:menuItem value="Menu 1"></ice:menuItem>
          <ice:menuItem value="Menu 2"></ice:menuItem>
          <ice:menuItem value="Menu 3"></ice:menuItem>
        </ice:menuBar>
      </ice:form>
    </ui:insert></td>
  </tr>
  <tr>
    <td><ui:insert name="contenu">SVP information content</ui:insert></td>
  </tr>
  <tr>
    <td><ui:insert name="footer"><ice:outputText value="#169; 2011 Objis"></ice:outputText></ui:insert></td>
  </tr>
</table>
</body>
</html>
```

# Démo Facelet avec IceFaces : Page v0



```
<!-- page-template.xhtml -->
<!-- *IceFacesPage1.xhtml -->
<!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:ui="http://java.sun.com/jsf/facelets"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:ice="http://www.icesoft.com/icefaces/component">
<head>
  <title><ui:insert name="title">Default title</ui:insert></title>
</head>
<body>
  <ui:composition template="">
    </ui:composition>
</body>
</html>
```

The screenshot shows an IDE with two tabs: 'page-template.xhtml' and '\*IceFacesPage1.xhtml'. The main editor displays the XHTML code for 'IceFacesPage1.xhtml'. The code includes XML namespaces for XHTML, JSF Facelets, JSF Core, JSF HTML, and IceFaces. It features a title tag with a facelet insert and a body containing a facelet composition tag. A small window on the right shows the project structure with files '/IceFacesPage1.xhtml' and '/page-template.xhtml'.

# Démo Facelet avec IceFaces :

## Page v1



```
<?xml version="1.0" encoding="UTF-8"?>
<!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:ui="http://java.sun.com/jsf/facelets"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:ice="http://www.icesoft.com/icefaces/component">
  <head>
    <title>
      <ui:insert name="title">Default title</ui:insert>
    </title>
  </head>
  <body>
    <ui:composition template="/page-template.xhtml">
      <div id="contenu">
        <ice:form>
          <ice:outputText value="Bienvenue dans ce premier ecran JSF Facelet"></ice:outputText>
        </ice:form>
      </div>
    </ui:composition>
  </body>
</html>
```

# Démo Facelet avec IceFaces :

## Page v1 rendu



# Démo Facelet avec IceFaces :

## Page v2

```
ICEfacesPage1.xhtml X
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transi
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:ui="http://java.sun.com/jsf/facelets"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:ice="http://www.icesoft.com/icefaces/component">
  <body>
    <ui:composition template="/page-template.xhtml">
      <ui:define name="title">
        Formation JSF avec Objis.
      </ui:define>
      <ui:define name="contenu">
        <ice:outputText value="Bienvenue dans ce premier ecran JSF Facelet"></ice:outputText>
      </ui:define>
    </ui:composition>
  </body>
</html>
```

# Démo Facelet avec IceFaces :

## Page v2 rendu





# Création listener personnalisé

- Pour écouter et traiter 3 types d'évènements
  - Type 1 : actions utilisateur
    - Action event déclenché par bouton ou lien
  - Type 2 : changement de valeur champ
    - `inputText`
  - Type 3 : phases JSF
    - Une des 6 phases traitement requête.
- Accès par programmation à l'évènement (`ActionEvent`) ou au contexte JSF (`FacesContext`)

# Comprendre un ActionEvent

## Method Summary

boolean	<a href="#"><b>isAppropriateListener</b></a> ( <a href="#">FacesListener</a> listener) Return true if this <a href="#">FacesListener</a> is an instance of a listener class that this event supports.
void	<a href="#"><b>processListener</b></a> ( <a href="#">FacesListener</a> listener) Broadcast this <a href="#">FacesEvent</a> to the specified <a href="#">FacesListener</a> , by whatever mechanism is appropriate.

## Methods inherited from class javax.faces.event.[FacesEvent](#)

[getComponent](#), [getPhaseId](#), [queue](#), [setPhaseId](#)

## Methods inherited from class java.util.EventObject

[getSource](#), [toString](#)

# Création listener

## type 1 : action utilisateur

1

```
<h:commandButton value = "Submit">
  <f:actionListener type = "mylisteners.SubmitAction"/>
</h:commandButton>

<h:commandLink value = "Click Me">
  <f:actionListener type = "mylisteners.SubmitAction"/>
</h:commandLink>
```

2

```
monMistener.java X
1 package com.objis.demosf.listeners;
2
3 import javax.faces.event.AbortProcessingException;
4 import javax.faces.event.ActionEvent;
5 import javax.faces.event.ActionListener;
6
7 public class monMistener implements ActionListener{
8
9     @Override
10    public void processAction(ActionEvent arg0) throws AbortProcessingException {
11        // TODO Auto-generated method stub
12
13    }
14}
```

# Création listener

## type 2 : changement valeur champ

1

```
<h:inputTextField>  
    <f:valueChangeListener type = "listeners.MyValueChangeListener"/>  
</h:inputTextField>
```

2

```
monListenerChangeValue.java X  
1 package com.objis.demosf.listeners;  
2  
3 import javax.faces.event.ValueChangeEvent;  
4  
5 public class monListenerChangeValue {  
6  
7     public void processValueChange(ValueChangeEvent vcEvent) {  
8  
9         String ancienneValeur = (String)vcEvent.getOldValue();  
10        String nouvelleValeur = (String)vcEvent.getNewValue();  
11  
12        // Travailler avec les valeurs  
13        System.out.println("Valeurs : " + ancienneValeur + " et " + nouvelleValeur);  
14    }  
15 }  
16
```

# Création listener type 3 : phases JSF

```
<lifecycle>
  <phase-listener>echo.listeners.PhaseListener</phase-listener>
</lifecycle>
</faces-config>
```

1

```
<f:view>
  <f:PhaseListener type = "mylisteners.MyPhaseListener"/>
</f:view>
```

PhaseListener.java X

```
package echo.listeners;

import javax.faces.event.PhaseEvent;
import javax.faces.event.PhaseId;

public class PhaseListener implements javax.faces.event.PhaseListener {

    @Override
    public void afterPhase(PhaseEvent event) {
        // Que faire juste après la phase ?
        event.getFacesContext().getExternalContext().log("APRES "+event.getPhaseId());
    }

    @Override
    public void beforePhase(PhaseEvent event) {
        // Que faire juste après la phase ?
        event.getFacesContext().getExternalContext().log("AVANT "+event.getPhaseId());
    }

    @Override
    public PhaseId getPhaseId() {
        // A quelle phase appeller ce listener ?
        return PhaseId.ANY_PHASE;
    }
}
```

2

# Création listener

## type 3 : exemple logs phases JSF

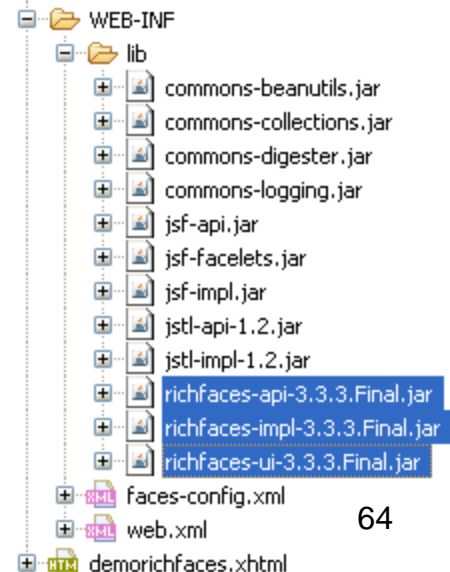
```
INFO: AVANT RESTORE_VIEW 1
4 janvier 2010 02:37:26 org.apache.catalina.core.ApplicationContext log
INFO: APRES RESTORE_VIEW 1
4 janvier 2010 02:37:26 org.apache.catalina.core.ApplicationContext log
INFO: AVANT APPLY_REQUEST_VALUES 2
4 janvier 2010 02:37:26 org.apache.catalina.core.ApplicationContext log
INFO: APRES APPLY_REQUEST_VALUES 2
4 janvier 2010 02:37:26 org.apache.catalina.core.ApplicationContext log
INFO: AVANT PROCESS_VALIDATIONS 3
4 janvier 2010 02:37:26 org.apache.catalina.core.ApplicationContext log
INFO: APRES PROCESS_VALIDATIONS 3
4 janvier 2010 02:37:26 org.apache.catalina.core.ApplicationContext log
INFO: AVANT UPDATE_MODEL_VALUES 4
4 janvier 2010 02:37:26 org.apache.catalina.core.ApplicationContext log
INFO: APRES UPDATE_MODEL_VALUES 4
4 janvier 2010 02:37:26 org.apache.catalina.core.ApplicationContext log
INFO: AVANT INVOKE_APPLICATION 5
4 janvier 2010 02:37:26 org.apache.catalina.core.ApplicationContext log
INFO: APRES INVOKE_APPLICATION 5
4 janvier 2010 02:37:26 org.apache.catalina.core.ApplicationContext log
INFO: AVANT RENDER_RESPONSE 6
4 janvier 2010 02:37:26 org.apache.catalina.core.ApplicationContext log
INFO: APRES RENDER_RESPONSE 6
```

# JSF et Ajax

- **Ajax4jsf gère arbre de composants tout au long de la requête / réponse Ajax.**

# RichFaces

- **Bibliothèque composants graphiques JSF**
  - Basé sur Ajax4jsf (+ ajout skins + composants)
  - `<a4j:>` et `<rich:>`
  - + de 100 composants visuels
  - demos : [jboss.org/richfaces/demos.html](http://jboss.org/richfaces/demos.html)
- **JSF 1.2 / RichFace 3.x**
  - RI Mojarra
  - MyFaces 1.2
- **JSF 2 / RichFaces 4**





# Composants a4j : <a4j:support>

- Permet d'ajouter le support d'ajax aux composants JSF de base

```
<h:commandButton value="Click">  
    <a4j:support event="onClick" action="#{bean.monAction}"  
        actionListener="#{bean.monActionListener}" reRender="text,panel"></a4j:support>  
</h:commandButton>
```

- Attribut '**event**' : évènement javascript qui lance la requête ajax
- Attribut '**reRender**' : contient les ids des composants à mettre à jour lors du retour de la réponse
- Attribut '**actionListener**' : binding de la méthode qui prend un `ActionEvent` en paramètre et retourne un type void.
- Attribut '**action**' : binding de la méthode qui invoque l'action de l'application

# Attributs commun A4J / RichFaces

- reRender
- ajaxSingle : booléen pour limiter les phases 2,3,4 seulement au composant qui envoie la requête
- Immediate : comme en jsf, pour éviter phase validation/conversion
- bypassUpdates : permet d'éviter phases MAJ modèle + invocation application pour passer directement à phase rendu réponse.
- onComplete : code à exécuter coté client à la fin de la requête
- limitToList : si valeur est true, la mise à jour des composants coté clients se limite à la liste dans l'attribut reRender
- timeOut : Max durée requête ajax
- Data : permet de récupérer du serveur une donnée durant la requête ajax. Ex : propriété d'un bean avec EL (sérialisée via JSON)

# Intégration JSF / RichFaces

1

Filtre RichFaces

```
<filter>
    <display-name>RichFaces Filter</display-name>
    <filter-name>richfaces</filter-name>
    <filter-class>org.ajax4jsf.Filter</filter-class>
</filter>
<filter-mapping>
    <filter-name>richfaces</filter-name>
    <servlet-name>Faces Servlet</servlet-name>
    <dispatcher>REQUEST</dispatcher>
    <dispatcher>FORWARD</dispatcher>
    <dispatcher>INCLUDE</dispatcher>
</filter-mapping>
```

2

```
<context-param>
    <param-name>org.ajax4jsf.VIEW_HANDLERS</param-name>
    <param-value>com.sun.facelets.FaceletViewHandler</param-value>
</context-param>
<context-param>
    <param-name>javax.faces.DEFAULT_SUFFIX</param-name>
    <param-value>.xhtml</param-value>
</context-param>
```

3

Canal : intégration avec Faces

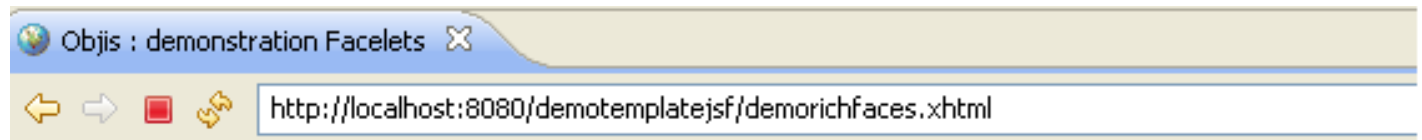
# Intégration JSF / RichFaces

```
demorichfaces.jsp X
7<head>
8<title>Démo RichFeces Objis</title>
9</head>
10<body>
11<rich:panel style="border:0;width:60%;text-align:center">
12    <h:form>
13        <rich:panel>
14            <f:facet name="header">
15                <h:outputText value="Exemple a4j" />
16            </f:facet>
17            <h:selectOneMenu value="#{person.name}">
18                <f:selectItem itemLabel="Jean" itemValue="Jean" />
19                <f:selectItem itemLabel="Stephane" itemValue="Stephane" />
20                <f:selectItem itemLabel="Michel" itemValue="Michel" />
21                <a4j:support event="onclick" reRender="text">
22                </a4j:support>
23            </h:selectOneMenu>
24            <h:outputText value="Selected name: #{person.name}" id="text"
25                style="font-weight:bold;" />
26        </rich:panel>
27    </h:form>
28</rich:panel>
29</body>
```

# Intégration Facelet / RichFaces

```
template.xhtml
1<html xmlns="http://www.w3.org/1999/xhtml"
2  xmlns:ui="http://java.sun.com/jsf/facelets"
3  xmlns:h="http://java.sun.com/jsf/html"
4  xmlns:f="http://java.sun.com/jsf/core"
5  xmlns:a4j="http://richfaces.org/a4j"
6  xmlns:rich="http://richfaces.org/rich">
7<ui:composition template="./WEB-INF/templates/template.xhtml">
8
9  <ui:define name="title">Richfaces test</ui:define>
10 <ui:define name="body">
11
12     <rich:panel style="border:0;width:60%;text-align:center">
13         <h:form>
14
15             <rich:panel>
16                 <f:facet name="header">
17                     <h:outputText value=" Exemple a4j" />
18                 </f:facet>
19                 <h:selectOneMenu value="#{person.name}">
20                     <f:selectItem itemLabel="Pierre" itemValue="Pierre" />
21                     <f:selectItem itemLabel="Paul" itemValue="Paul" />
22                     <f:selectItem itemLabel="Jacques" itemValue="Jacques" />
23                     <a4j:support event="onclick" reRender="text">
24                         </a4j:support>
25                 </h:selectOneMenu>
26
27                 <h:outputText value="Selected name: #{person.name}" id="text"
28                     style="font-weight:bold;" />
29             </rich:panel>
30         </h:form>
31     </rich:panel>
32 </ui:define>
33</ui:composition>
34</html>
```

# Intégration Facelet / RichFaces



## Richfaces test

Exemple a4j

Jacques ▼ Votre choix : Jacques

# Modification Skins

- Skins par défaut dans META-INF/skins de l'archive RichFaces-impl-xxx.jar
  - blueSky, Laguna, classic, glass-x,...

```
50      <!-- INFO : pour skin perso, modifier le Skin (Fichiers dans META-INF/skins/)-->
51
52      <context-param>
53          <param-name>org.richfaces.SKIN</param-name>
54          <param-value>blueSky</param-value>
55      </context-param>
```

Web.xml

- Pour personnaliser skin,
  - désarchiver richfaces-impl-xxx.jar
  - Sur un fichier .properties, faire modif css désirées
  - Archiver à nouveau le jar et mettre ds projet

# Concepts clés RichFaces

- **Concept N°1 : envoi d'une requête Ajax**
  - `<a4j:commandLink>`, `<a4j:commandButton>`
  - `<a4j:support>`
  - `<a4j:poll>`
  - Attribut `limitToList`
- **Concept N°2 : mise à jour partielle de la page**
  - Attribut `reRender`
  - `<a4j:outputPanel>`
- **Concept N°3 : zones à traiter côté serveur**
  - `<a4j:region>`
  - Attribut `AjaxSingle`
  - Attribut `process`



# Composants clés RichFaces

## ➤ Composants d'entrée

- `<rich:inplaceInput>`, `<rich:inplaceSelect>`
- `<rich:suggestionBox>`, `<rich:comboBox>`,...

## ➤ Composants de sortie

- `<rich:panel>`, `<rich:simpleTogglepanel>`
- `<rich:tabPanel>`, `<rich:modalPanel>`, `<rich:PanelBar>`

## ➤ Composants d'itération de données

- `<rich:dataTable>`, `<rich:dataList>`, `<rich:dataGrid>`
- `<rich:dataScroller>` (Pagination)

## ➤ Composants de sélection

- `<rich:pickList>`, `<rich:orderingList>`

## ➤ Composants de Menu

- `<rich:dropDownMenu>`, `<rich:contextMenu>`

## ➤ Données et arbres scrollables

- `<rich:scrollableDataTable>`, `<rich:tree>`

# Concept N°1 : envoi requête Ajax

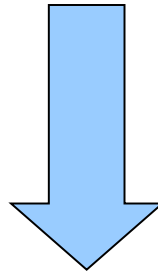
```
<h:form>
  <h:panelGrid columns="3">
    <h:outputText value="Age:" />
    <h:inputText value="#{userBean.age}" size="4" />
    <a4j:commandButton value="Entrez Age" reRender="age" />
  </h:panelGrid>
  <h:panelGrid>
    <h:outputText id="age" value="Votre age: #{userBean.age}" />
  </h:panelGrid>
</h:form>
```

```
public class UserBean {
    Integer age;

    public UserBean() {
    }

    public Integer getAge() {
        return age;
    }

    public void setAge(Integer age) {
        this.age = age;
    }
}
```



Age:

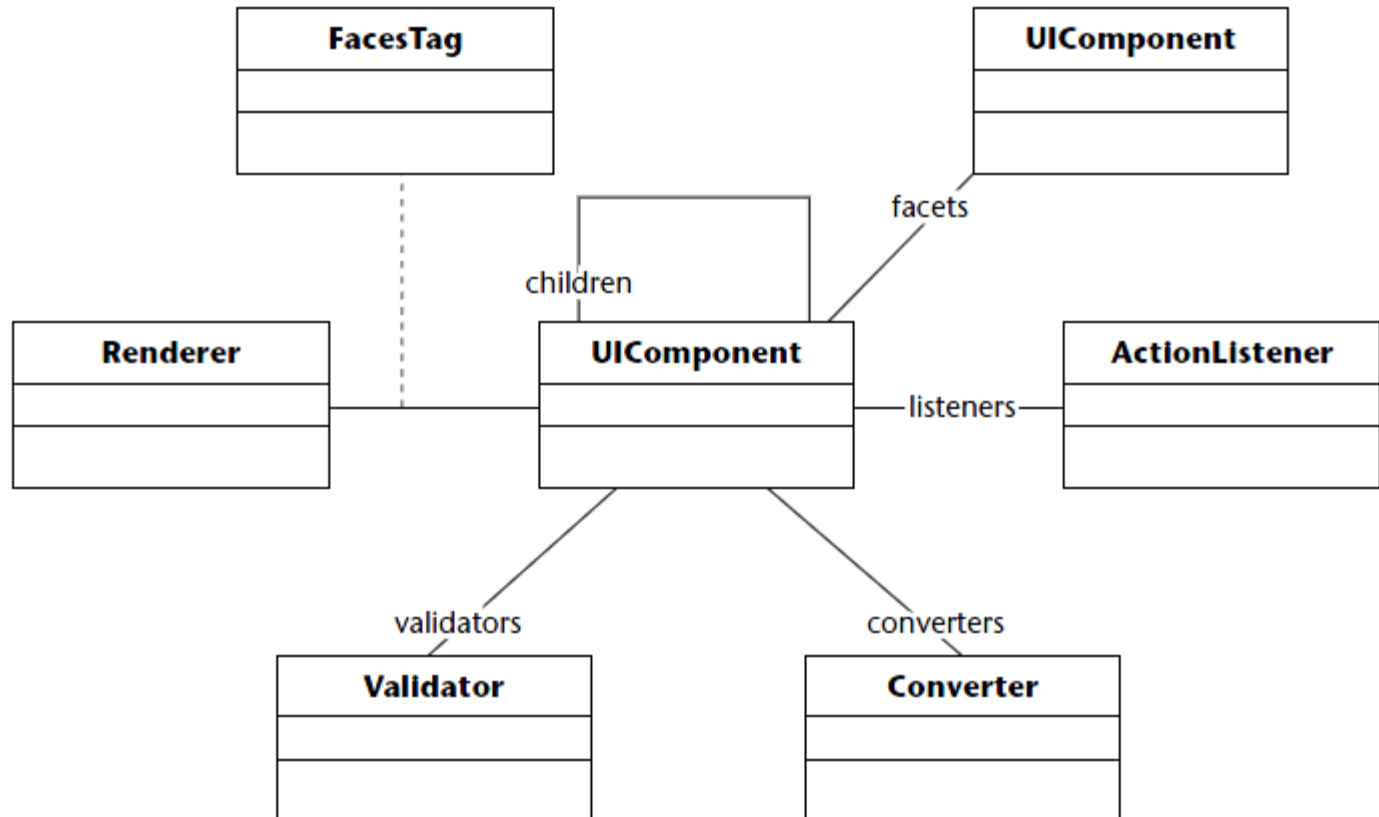
Votre age: 33

# Création composant personnalisé

- 1) Créer classe hérite de **javax.faces.component.UIComponentBase**
  - Redéfinir ma méthode `encodeBegin(FaceContext context)`
  - Si attribut, alors redéfinir `encodeEnd()`
- 2) Enregistrer la classe dans **faces-config.xml**
  - balise **<component>**
- 3) **Créer un Tag** spécifique (classe héritant de `UIComponentTag`)
  - Ex : `sf:sellerHello`
- 4) **Créer le descripteur (TLD) du Tag**

[www.theserverside.com/news/1364786/Building-Custom-JSF](http://www.theserverside.com/news/1364786/Building-Custom-JSF)

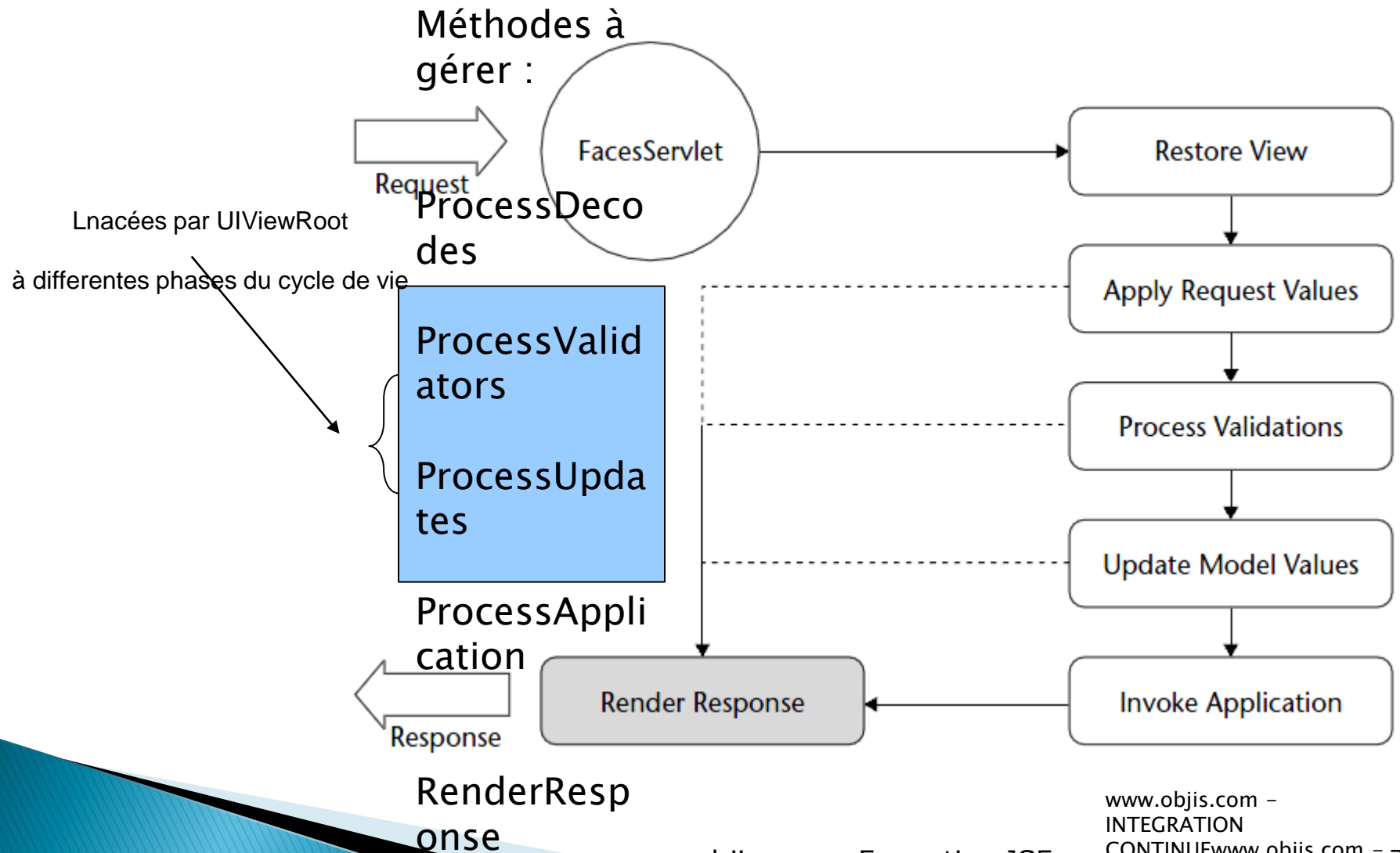
# Composant personnalisé : Comprendre son environnement



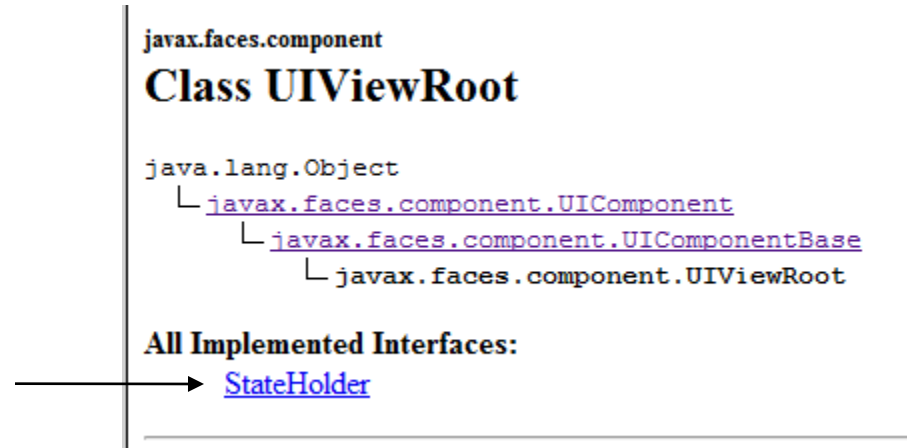
Java Resources: src

- com.objjis.demosf.component
  - ObjisUIScroller.java
- com.objjis.demosf.component.renderer
  - ObjisScrollerRenderer.java
- com.objjis.demosf.component.taglib
  - ObjisScrollerTag.java
- com.objjis.demosf.component.util

# Composant personnalisé : Comprendre les sollicitations



# Composant personnalisé : Comprendre UIViewRoot



```
public class UIViewRoot
extends UIComponentBase
```

**UIViewRoot** is the **UIComponent** that represents the root of the **UIComponent** tree. This component has no rendering, it just serves as the root of the component tree, and as a place to hang per-view [PhaseListeners](#).

For each of the following lifecycle phase methods:

- [processDecodes\(javax.faces.context.FacesContext\)](#)
- [processValidators\(javax.faces.context.FacesContext\)](#)
- [processUpdates\(javax.faces.context.FacesContext\)](#)
- [processApplication\(javax.faces.context.FacesContext\)](#)
- RenderResponse, via [encodeBegin\(javax.faces.context.FacesContext\)](#) and [encodeEnd\(javax.faces.context.FacesContext\)](#)

# Impact à chaque phase

- Phase 1 (Restauration de la vue)
  - Gestion de l'état (coté client ou serveur) : sauver / restaurer
  - (optionnel) Implémenter `javax.faces.component.StateHolder`
    - Ex : `saveState` retourne tableau d'objets Sérialisables
    - Ex : `restoreState` attend un tableau d'objets Sérialisables
- Phase 2 (Traitement de la requête)
  - 'Décoder' (exploiter en interne) les paramètres de requête
    - Le faire dans composant OU déléguer à classe (Renderer)
  - Impulsion par méthode `processDecode` du `UIViewRoot`
  - (obligatoire) Implémenter méthode `decode()`
  - `Decode()` travailler avec le `Renderer`

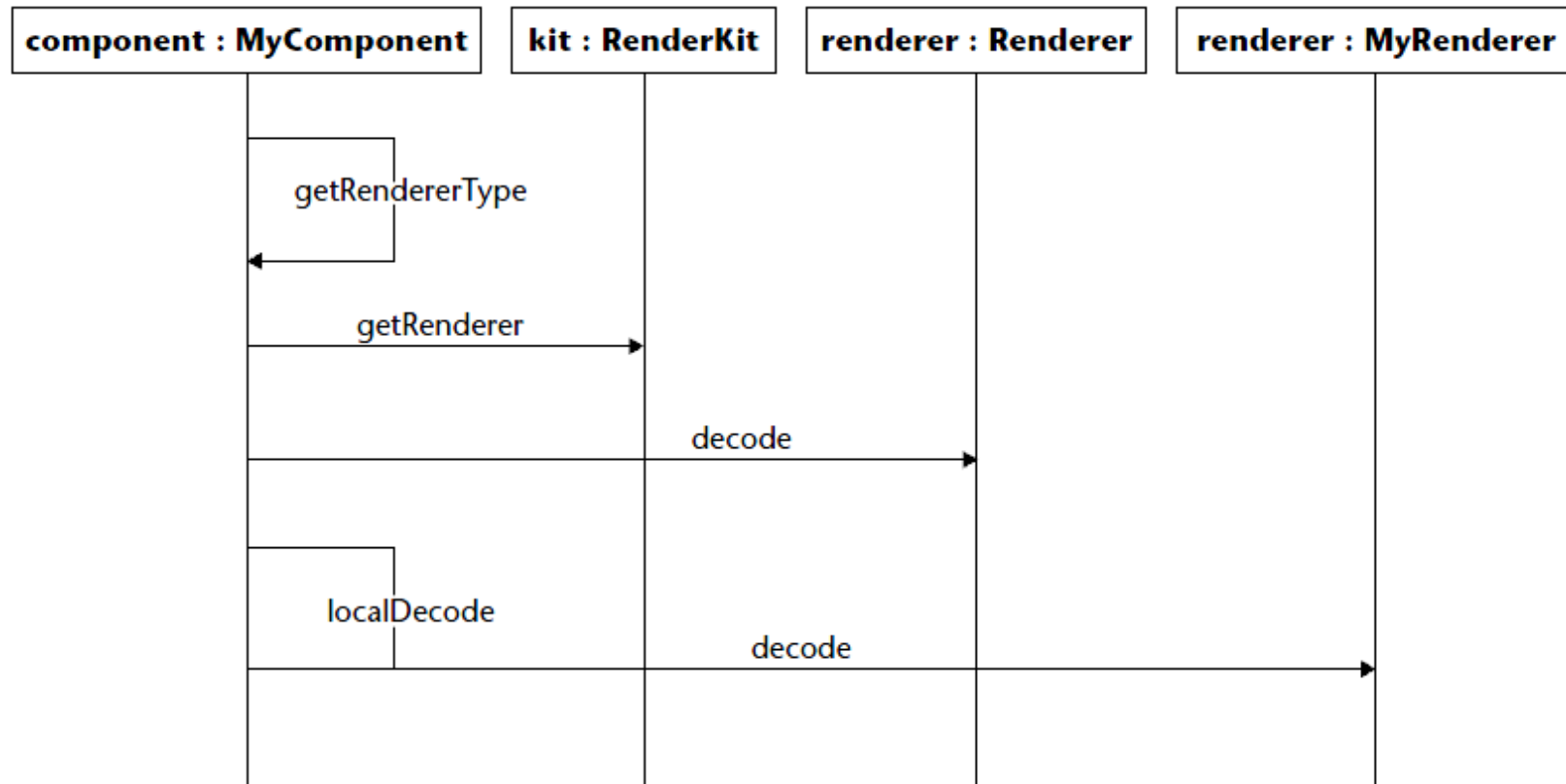
# Création composant personnalisé

- 1) Créer classe hérite de **javax.faces.component.UIComponentBase**
  - Redéfinir ma méthode `encodeBegin(FaceContext context)`
  - Si attribut, alors redéfinir `encodeEnd()`
- 2) Enregistrer la classe dans **faces-config.xml**
  - balise **<component>**
- 3) **Créer un Tag** spécifique (classe héritant de `UIComponentTag`)
  - Ex : `sf:sellerHello`
- 4) **Créer le descripteur (TLD) du Tag**

[www.theserverside.com/news/1364786/Building-Custom-JSF](http://www.theserverside.com/news/1364786/Building-Custom-JSF)



# Composant personnalisé : Rendu visuel (Render)



# Composant personnalisé : decode() dans composant

```
ObjisUIScroller.java X ObjisScrollerRenderer.java

public void decode(FacesContext context) {
    logger.info("Décodage");

    if (context == null) {
        NullPointerException npe = new NullPointerException(
            "FacesContext null !!! " + getId());
        logger.error("ECHEC decodage pourr composant ObjisUIScroller");
        throw npe;
    }
    if (getRendererType() == null) {
        Renderer renderer = getSharedRenderer(context); // APPEL DE VOTRE RENDER
        renderer.decode(context, this);
    } else {
        super.decode(context); // render kit render
    }
}

private Renderer getSharedRenderer(FacesContext context) {
    RenderKitFactory rkFactory = (RenderKitFactory) FactoryFinder
        .getFactory(RenderKitFactory.HTML_BASIC_RENDER_KIT);
    RenderKit renderKit = rkFactory.getRenderKit(context, context
        .getViewRoot().getRenderKitId());
    return renderKit.getRenderer(ObjisUIScroller.FAMILY,
        ObjisScrollerRenderer.TYPE);
}
```

# Composant personnalisé : decode() dans Renderer perso

```
ObjisUIScroller.java  ObjisScrollerRenderer.java X

private void decodeCommandName(FacesContext context, ObjisUIScroller scroller) {

    String clientId = scroller.getClientId(context);
    String rightClientId = getRightClientId(clientId);
    String leftClientId = getLeftClientId(clientId);
    Map requestParameterMap = context.getExternalContext().getRequestParameterMap();
    String value = (String) requestParameterMap.get(clientId);

    String rightValue = (String) requestParameterMap.get(rightClientId);
    String leftValue = (String) requestParameterMap.get(leftClientId);

    String commandName = null;
    logger.debug("rightValue = " + rightValue);
    logger.debug("leftValue = " + leftValue);

    if (null != rightValue && clientId.equals(rightValue)) {
        scroller.performRightAction(context);
        commandName = rightValue;
    } else if (null != leftValue && clientId.equals(leftValue)) {
        scroller.performLeftAction(context);
        commandName = leftValue;
    } else {
        throw new IllegalStateException(
            "no valid value returned to decode:" + " rightValue = "
            + rightValue + " leftValue = " + leftValue
            + " clientId = " + clientId);
    }
}
```

# Composant personnalisé : encodeBegin()

```
public void encodeBegin(FacesContext context, UIComponent component) throws IOException {
    checkState(context, component);
    if (!component.isRendered()) {
        return;
    }
    ObjisUIScroller scroller = (ObjisUIScroller) component;
    UIForm form = ComponentTreeUtils.getParentForm(scroller);
    if (null != form) {
        String formName = getFormName(form);
        String clientId = scroller.getClientId(context);
        String rightClientId = getRightClientId(clientId);
        String leftClientId = getLeftClientId(clientId);
        List<Element> elements = new ArrayList<Element>();
        // setup the left image
        elements.add(imgElement(scroller.getLeftScrollImg(), leftClientId,
                                clientId, formName));
        elements.add(selectedElement(scroller));
        // setup the right image
        elements.add(imgElement(scroller.getRightScrollImg(),
                                rightClientId, clientId, formName));
        elements.add(hiddenElement(rightClientId));
        elements.add(hiddenElement(leftClientId));
        output(elements, context);
    } else {
        queueMissingFormErrorMessage(context, scroller);
    }
}
```