

JSF 2

27/03/2017

Walid YAICH

walid.yaich@esprit.tn

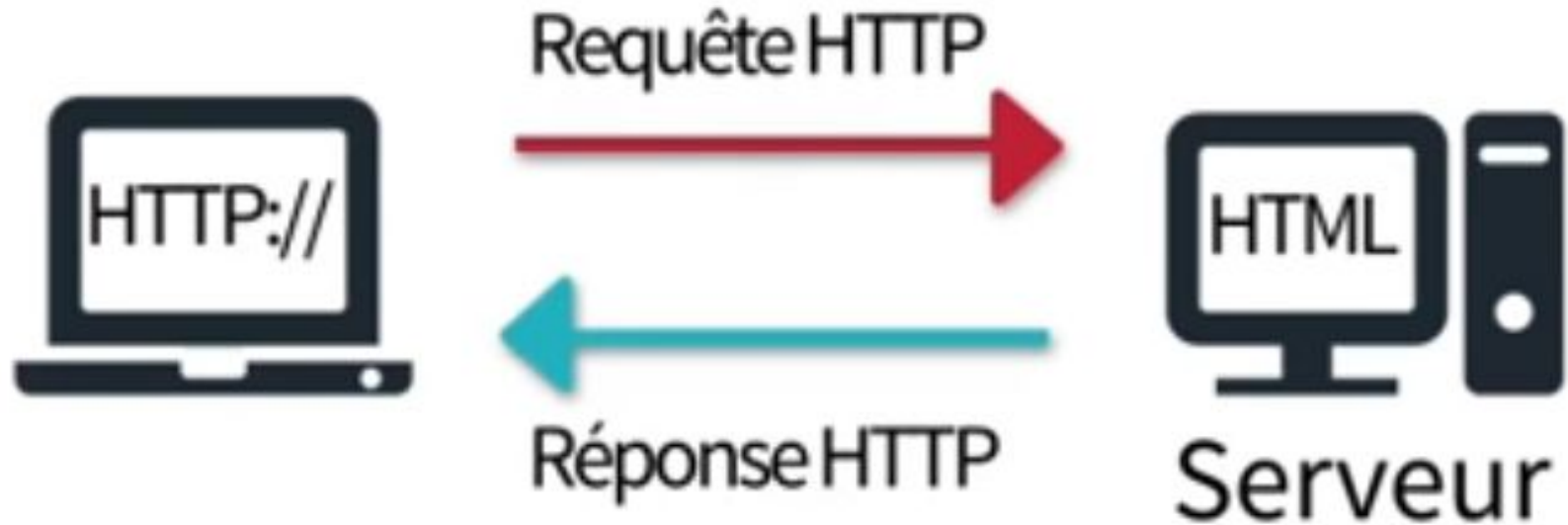
Bureau E204



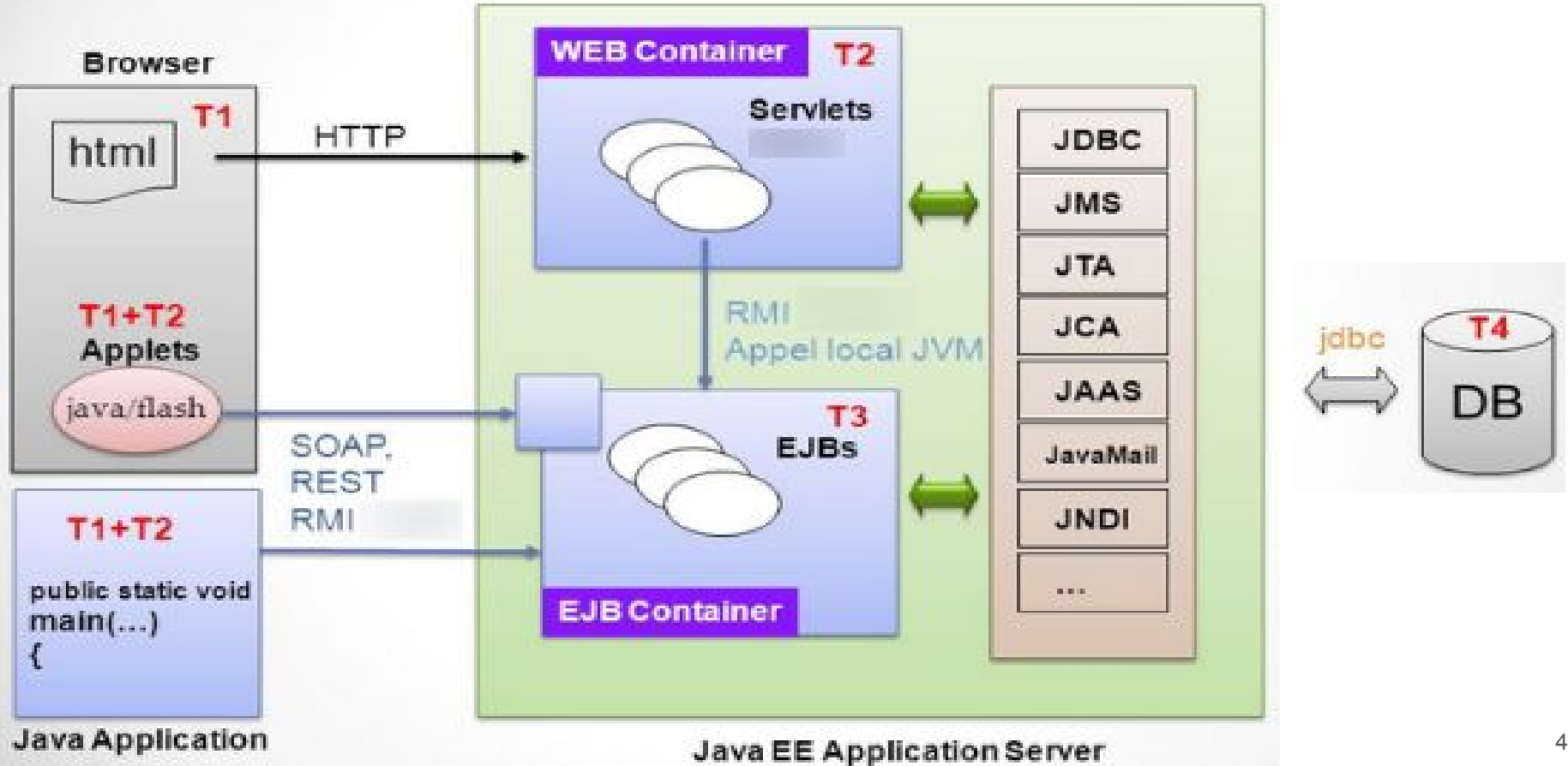
Plan

- HTTP - Architecture client serveur
- Architecture n-tiers
- Servlet
- JSP(JavaServer Pages)
- JSF (JavaServer Faces)
- Vue globale
- Exemple de Structure d'une application JSF
- autocomplete dans les pages XHTML
- Exemple de template
- Projet timesheet - login
- Test
- Troubleshooting

HTTP - Architecture client serveur

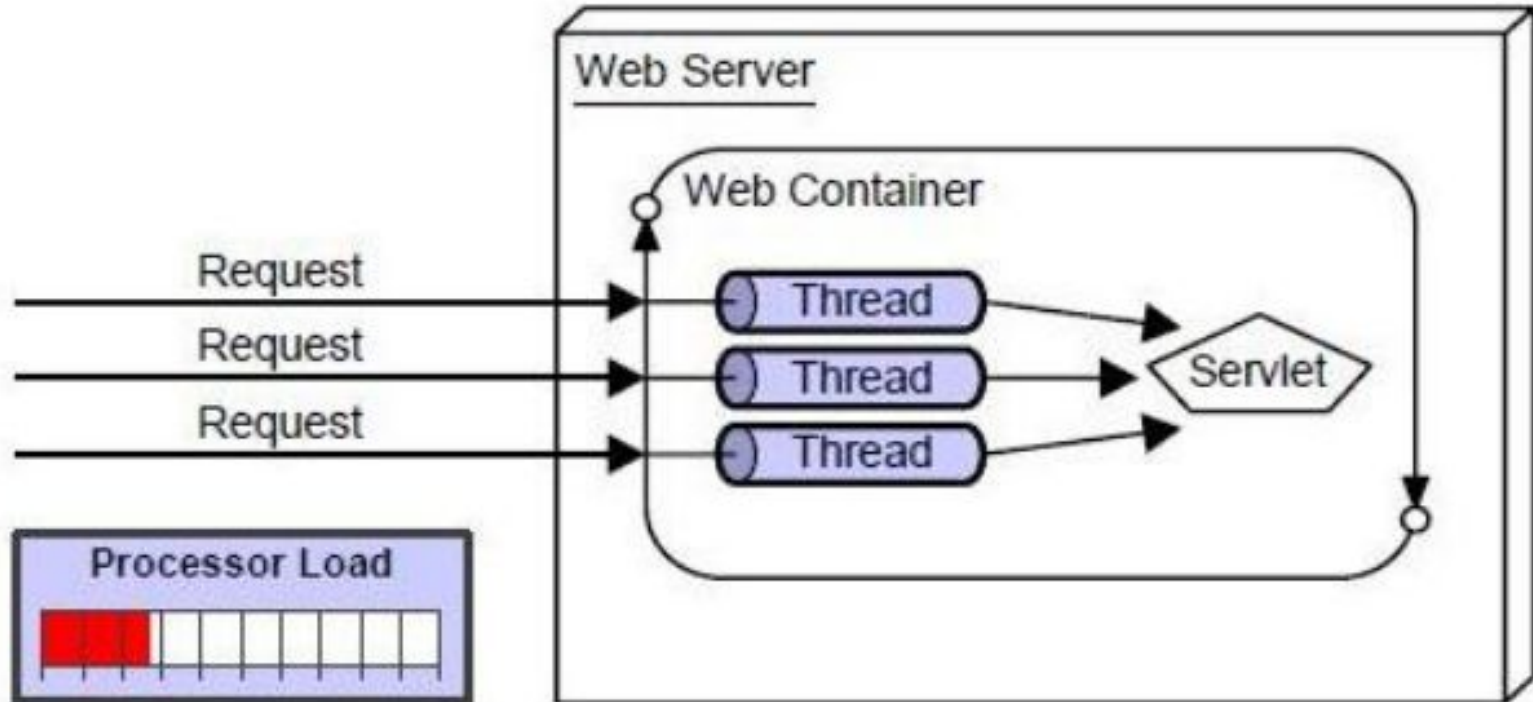


Architecture n-tiers



Servlet

Une servlet est une simple classe Java, qui a la particularité de **permettre le traitement de requêtes HTTP et la personnalisation de réponses HTTP**.



```
public Bonjour extends HttpServlet {
```

```
    public void doGet(HttpServletRequest request, HttpServletResponse response)  
    throws ServletException, IOException {
```

```
        response.setContentType("text/html") ;
```

```
        PrintWriter out = response.getWriter() ;
```

```
        out.println("<html>") ;
```

```
        out.println("<head>") ;
```

```
        out.println("<title>Bonjour le monde !</title>") ;
```

```
        out.println("</head>") ;
```

```
        out.println("<body>") ;
```

```
        out.println("<h1>Bonjour le monde !</h1>") ;
```

```
        out.println("</body>") ;
```

```
        out.println("</html>") ;
```

```
    }
```

```
    public void doPost(HttpServletRequest request, HttpServletResponse response)  
    throws ServletException, IOException {
```

```
        doGet(request, response) ;
```

```
    }
```

JSP(JavaServer Pages)

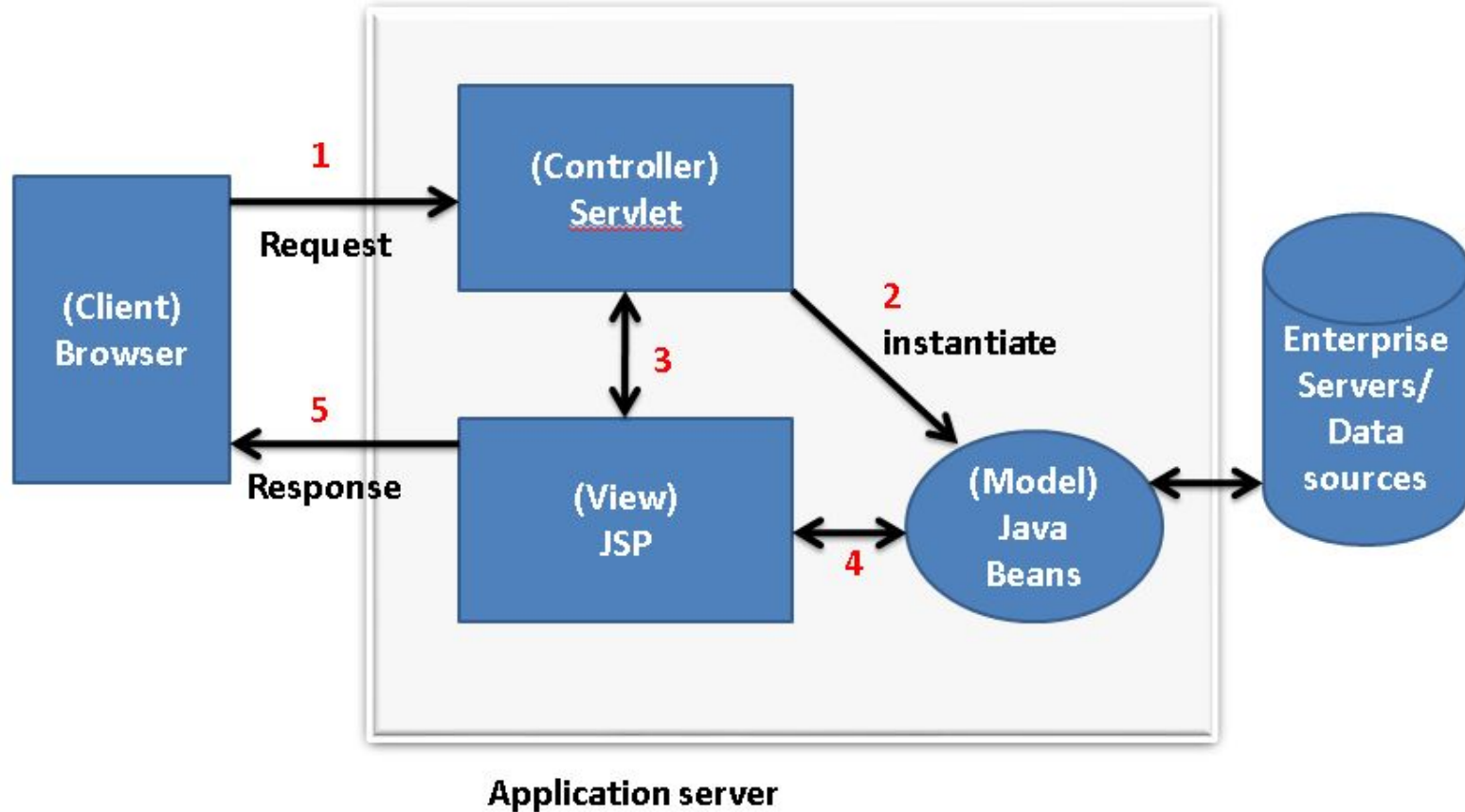
Le modèle des servlets était de trop **bas niveau** et c'est la raison pour laquelle, les pages JSP (JavaServer Pages) ont ensuite pris le relais.

Les JSP (Java Server Pages) sont une technologie Java qui permet la génération de pages **web dynamiques**.

La version 1.0 est sortie en Juin 1999

```
<HTML>
<HEAD>
<TITLE>Test</TITLE>
</HEAD>
<BODY>
<%!
int minimum(int val1, int val2) {
    if (val1 < val2) return val1;
    else return val2;
}
%>
<% int petit = minimum(5,3);%>
<p>Le plus petit de 5 et 3 est <%= petit %></p>
</BODY>
</HTML>
```

JSP (JavaServer Pages) - MVC 1

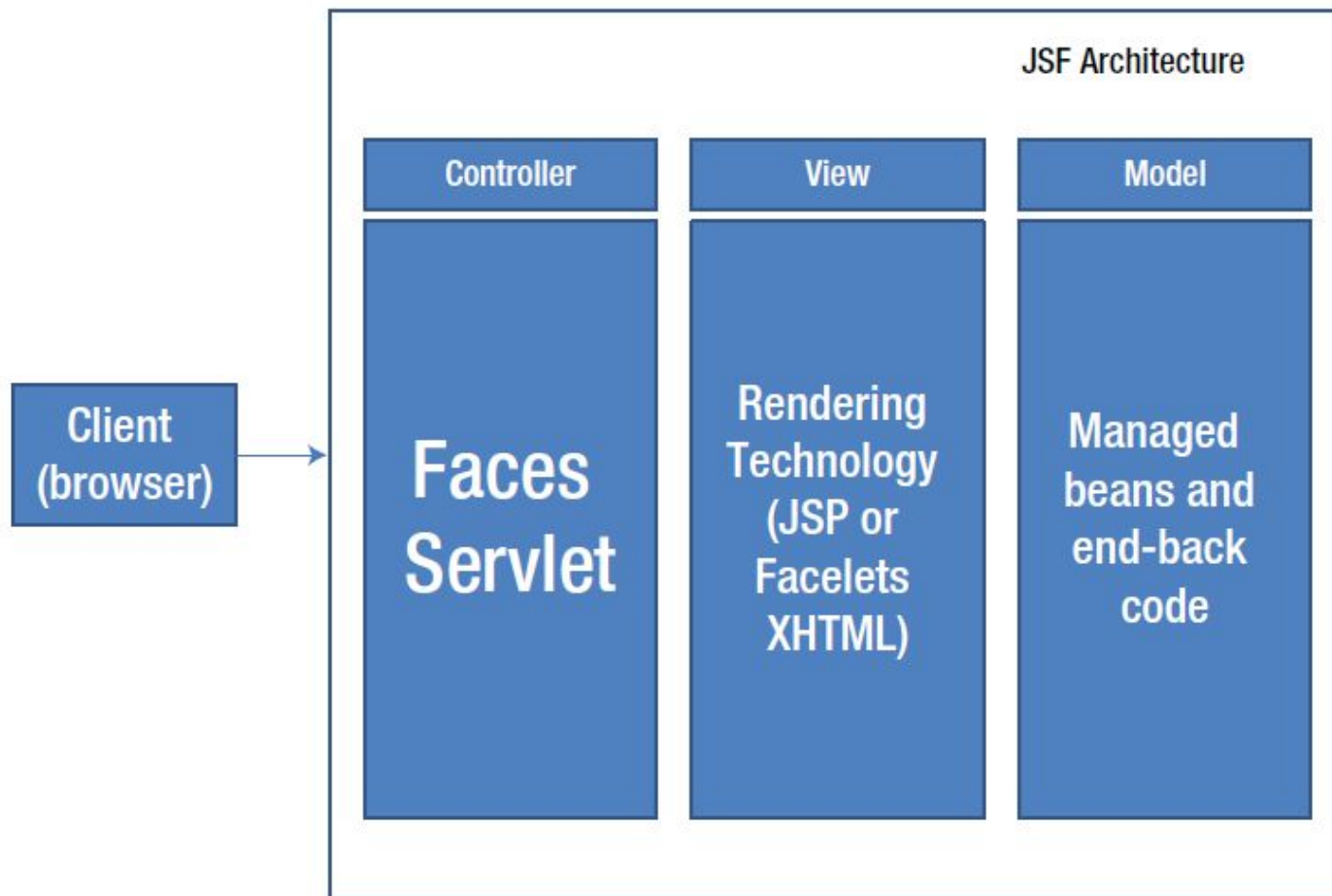


JSF (JavaServer Faces)

JSF est une technologie utilisée côté serveur dont le but est de faciliter le développement de l'interface utilisateur en séparant clairement la partie « interface » de la partie « métier ».

- La [spécification](#) JSF définit le mode de fonctionnement du framework.
- JSF possède une [implémentation](#) de référence qui s'appelle mojarra.
- La version 1.0 de Java Server Faces, développée sous la JSR-127, a été validée en mars 2004.
- La version stable JSF 2.2 est sortie en 2013.

JSF (JavaServer Faces) - MVC 2

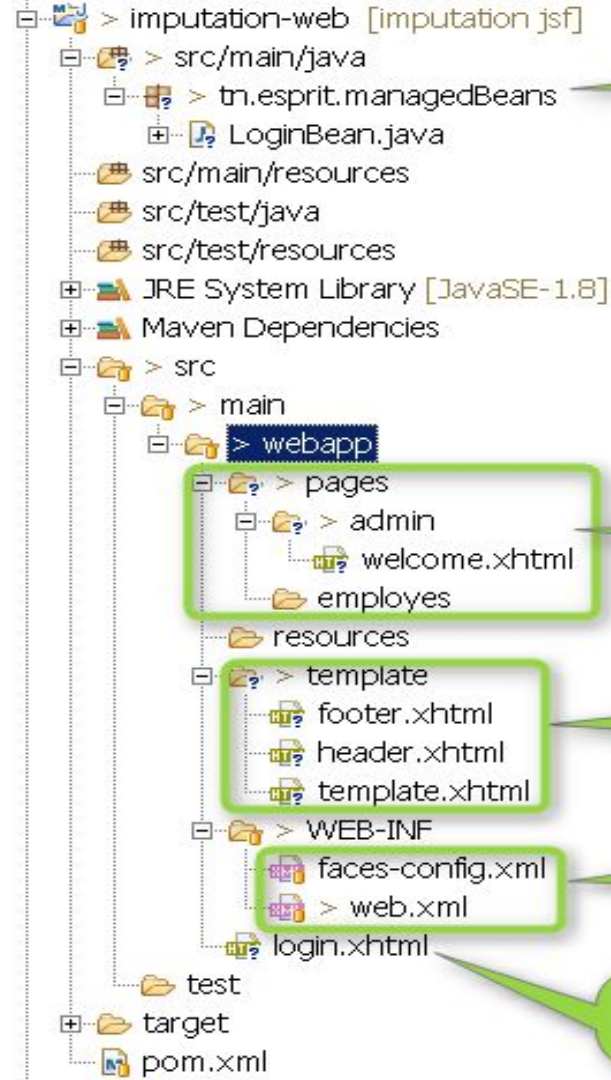


JSF (JavaServer Faces) - MVC 2

Controller : Lorsque l'utilisateur clique sur un bouton, par exemple, ceci provoque l'envoi d'une requête HTTP, celle-ci est interceptée par FacesServlet, qui examine la requête et exécute différentes actions sur le modèle à l'aide de beans gérés(Managed Bean).

Model : c'est les managed bean et tous le code backend.

View : C'est la partie qui s'occupe de la construction de la vue, depuis la version 2.0, c'est les facelets XHTML qui sont utilisé pour l'écriture des vues.



Les Managed Beans

Exemple de Structure d'une application JSF

Facelet pages

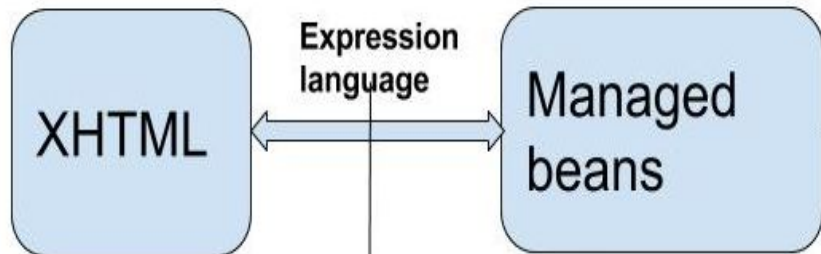
Le template de notre application web

Fichiers de configuration

La page de d'accueil

Vue globale

Conteneur web



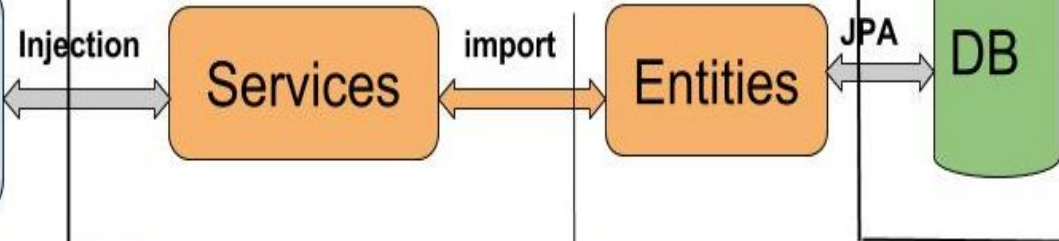
```
#{loginBean.doLogin()}
```

```
@ManagedBean
@SessionScoped
public class LoginBean {
    private String login;
    private String password;
    private Employe employe;
    private boolean loggedIn; // d

    // injection de dépendances
    @EJB
    EmployeeService employeeService;

    public String doLogin() {
```

Conteneur EJB



```
import tn.esprit.timesheet.entities.Contract;
```

```
@Stateless
public class EntrepriseService implements
    @PersistenceContext(unitName = "timesheet")
    EntityManager em;

    @Override
    public int ajouterEntreprise(
```

```
@Entity
public class Contrat implements Serializable {

    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int reference;

    @Temporal(TemporalType.DATE)
    private Date dateDebut;
```

EmployeeService.java

@Stateless

@LocalBean

public class EmployeeService **implements** EmployeeServiceRemote {

@Override

public Employee getEmployeeByEmailAndPassword(String email, String password) {

TypedQuery<Employee> query = em.createQuery("Select e from Employee e "
+ "where e.email=:email and "
+ "e.password=:password", Employee.class);

query.setParameter("email", email);
query.setParameter("password", password);

Employee employee = null;

try{

employee = query.getSingleResult();

}**catch**(NoResultException e){

logger.info("Aucun employe trouve avec email : " + email);

}

return employee;

}

@ManagedBean
@SessionScoped

LoginBean.java

```
public class LoginBean {  
    private String login;  
    private String password;  
    private Employee employee;  
    private boolean loggedIn; // default is false  
  
    // injection de dépendances  
    @EJB  
    EmployeeService employeeService;  
  
    public String doLogin() {  
        String navigateTo = "null";  
        employee = employeeService.getEmployeeByEmailAndPassword(login, password);  
  
        if (employee != null && employee.getRole() == Role.ADMIN) {  
            navigateTo = "/pages/admin/welcome?faces-redirect=true";  
            loggedIn = true;  
        } else {  
            FacesContext.getCurrentInstance().addMessage("form:btn", new FacesMessage("bad credentials"));  
        }  
        return navigateTo;  
    }  
}
```

logout() dans LoginBean.java

```
public String doLogout() {  
    FacesContext.getCurrentInstance().getExternalContext().invalidateSession();  
    return "/login?faces-redirect=true";  
}
```


autocomplete dans les pages XHTML

Properties for imputation-web

type filter text

- + Java Compiler
- + Java Editor
 - ... Javadoc Location
- + JavaScript
- + JAX-RS
- + JBoss Tools Knowledge Base
 - ... JSF Validation
 - ... JSP Fragment
- + Maven
 - ... Project Archives
- + **Project Facets**
- ... Project References
- ... Refactoring History
- ... Run/Debug Settings
- ... Server

Project Facets

Configuration: <custom>

Project Facet	Version	
<input type="checkbox"/> Application Client module	6.0	▼
+ <input type="checkbox"/> Axis2 Web Services		
<input type="checkbox"/> CDI (Contexts and Dependency Injection)	1.2	▼
<input type="checkbox"/> CXF 2.x Web Services	1.0	
<input checked="" type="checkbox"/> Dynamic Web Module	2.5	▼
<input type="checkbox"/> EAR	6.0	▼
<input type="checkbox"/> EJB Module	3.1	▼
<input type="checkbox"/> EJBDoclet (XDoclet)	1.2.3	▼
<input checked="" type="checkbox"/> Java	1.8	▼
<input checked="" type="checkbox"/> JavaScript	1.0	
<input checked="" type="checkbox"/> JavaServer Faces	2.2	▼
<input type="checkbox"/> JAX-RS (REST Web Services)	2.0	▼

autocomplete dans les pages XHTML

Ctrl + espace



`<h:commandButton action="log" value="" />`

`loginBean : LoginBean`

Exemple de template - template.xhtml

```
<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">

  <h:head>
    <ui:include src="/template/header.xhtml" />
    <title><ui:insert name="title"></ui:insert></title>
  </h:head>

  <body>
    <ui:insert name="content"></ui:insert>
    <ui:include src="/template/footer.xhtml" />
  </body>

</html>
```

<ui:insert> définit un point d'insertion dans un template, dans lequel on pourra ensuite insérer un contenu en utilisant **<ui:define>**

footer.xhtml

```
<ui:composition 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">
```

```
  <center>esprit 2017</center>
```

```
</ui:composition>
```

header.xhtml

```
<ui:composition 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">
```

```
  <center>Application Timesheet</center>
```

```
</ui:composition>
```

Projet timesheet - login.xhtml

```
<body>
    <h:form id="form">
        <h:panelGrid columns="2">
            1 <h:outputText value="login" />
            2 <h:inputText value="#{loginBean.login}" />
            3 <h:outputText value="password" />
            4 <h:inputSecret value="#{loginBean.password}" />
            5 <h:commandButton id="btn" action="#{loginBean.doLogin()}" value="login" />
            6 <h:message for="btn" />
                FacesContext.getCurrentInstance().addMessage("form:btn", new FacesMessage("bad credentials"));
        </h:panelGrid>
    </h:form>
</body>
```

Column 1	Column 2
login	
password	
login	bad credentials

Welcome.xhtml

```
<ui:composition 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"
  template="/template/template.xhtml">
```

```
<ui:define name="title">welcome</ui:define>
```

```
<ui:define name="content">
```

```
welcome #{loginBean.employe.nom} #{loginBean.employe.prenom}
```

```
<h:form>
```

```
    <h:commandButton action="#{loginBean.doLogout()}" value="logout" />
```

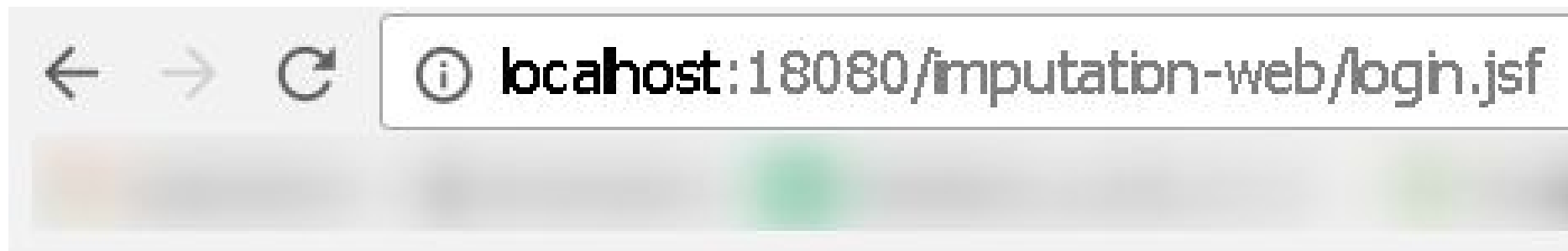
```
</h:form>
```

```
</ui:define>
```

```
</ui:composition>
```

Expression
Language

Test



login

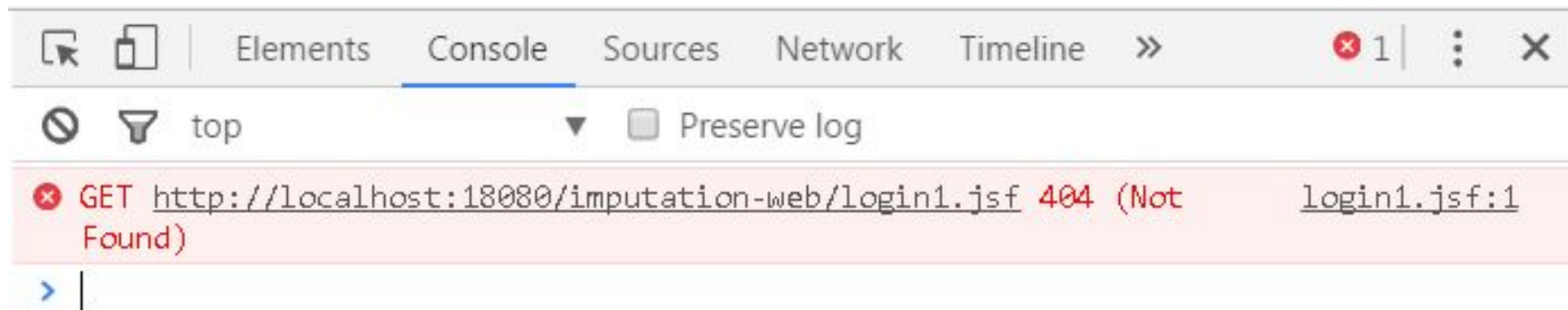
password

login

Troubleshooting



`/imputation-web/login1.jsp`



Troubleshooting

Vérifier que eclipse a bien déployer les différents dossiers et fichiers sous wildfly.

