<XML num="6"/>
Application Web en Java 3

- Emilien Bondu (Airbus Defence and Space)
- emilien.bondu@cassidian.com

Sommaire

- Cours 1 : Introduction à XML et validation (XSD)
- Cours 2 : Validation (DTD, Relax NG, Schematron) et transformation (XPath, XSL; XSL-FO)
- Cours 3 : Recherche XML (XQuery), Base de données XML, Liens XML, Manipulation XML en Java: Dom, Sax
- Cours 4 : Manipulation XML en Java : StAX, Data-binding, JavaEE : tomcat
- Cours 5 : Manipulation XML en JavaEE: servlet, JSP, ExpressionLanguage, TagLib
- Cours 6 : TagLib (suite), Spring, JSF, AJAX

Rappels

- Une Servlet:
  - permet d'étendre les services rendu par un serveur
  - traite directement les requêtes envoyées par les clients
  - possède un cycle de vie géré par le conteneur d'application
  - est un des composants web d'une application web JEE

Rappels

- Une Servlet :
  - est initialisée par Tomcat lors du déploiement
  - est une instance Java dont les attributs sont partagés entre les différents appels
  - est instanciée par le loader Tomcat conformément à un fichier web.xml
  - donne accès aux sessions associées aux requêtes reçues

Rappels

- Servlet et MVC
  - une servlet joue le rôle de contrôleur
  - le conteneur joue le rôle de contrôleur
  - une servlet manipule les objets métier de l'application pour les transmettre à la vue
  - écrire directement les données dans la réponse à l'aide du writer est prohibé.

Rappels

- Java Beans:
  - est une encapsulation des attributs d'un objet métier
  - est un objet sérialisable respectant des conventions de nommage
  - doit fournir un constructeur avec des attributs
  - propose des accesseurs aux attribut privés de l'objet

Rappels

- Entreprise Java Bean:
  - est un composant métier géré par un conteneur JEE exploitant les annotations Java
  - possède un cycle de vie mais est auto-suffisant
  - un sessionBean peu conserver ou non son état entre les appels
  - Un entityBean se charge d'assurer sa propre persistance dans une base de données

Rappels

- Expression Language :
  - permet d'accéder aux getters et setters des propriétés des JavaBeans
  - aide à se passer des scriptlets dans les pages JSP
  - fonctionne par réflexion
  - donne accès à des objets relatifs à la page, dans différents scopes

Rappels

- Tag Library :
  - se déclare dans une JSP en important un descripteur au format *.tld
  - s'utilise au moyen de balises XML et d'attributs appelant du code java
  - peut s'utiliser au sein d'une expression EL
  - permet d'étendre les actions utilisables au sein d'une page JSP

Application Web ***en Java***

<% taglib

JSP Tag Library

%>

exemple de taglib

- Java Standard Tag Library (JSTL)
  - core (base) : http://java.sun.com/jsp/jstl/core
  - Format : http://java.sun.com/jsp/jstl/fmt
  - XML : http://java.sun.com/jsp/jstl/xml
  - SQL : http://java.sun.com/jsp/jstl/sql
  - Fonctions : http://java.sun.com/jsp/jstl/functions

<% taglib

%>

Taglib Format <fmt:/>

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>
```

- Principales actions
  - gérer l'internationalisation **(i18n)**
    - afficher des messages dans la langue du client
    - gérer des bundle de langues
  - Formater et transformer
    - date
    - nombre

<% taglib

Taglib Format <fmt:/>

%>

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>
```

- Resource bundle
  - ensemble de *.properties contenant des ressources localisées
    - chaque fichier est suffixé par le code ISO de la langue
    - ensemble de clé/valeur

```
monbundle_fr.properties          message.title=Bonjour les Français
monbundle_en.properties          message.title=Hello English
…
monbundle_fr_FR.properties       message.title=Bonjour les canadiens
```

<% taglib

Taglib Format <fmt:/>

%>

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>
```

- Définir les bundles
  - **<fmt:setBundle />**
    - baseName (string) : nom du bundle (package.nomBundle)
    - var (string) : nom de la variable associée au bundle
    - scope (string) : scope où stocker la variable

```
<fmt:setBundle baseName="monbundle"/>

<fmt:setBundle baseName="monbundle_extended" var="extended"/>
```

Taglib Format <fmt:/>

%>

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>
```

- Définir / forcer la locale
  - **<fmt:setLocale />**
    - value (string/locale) : locale à utiliser
    - variant (string) : variante spécifique au système / navigateur
    - scope (string) : scope où stocker la locale

```
<fmt:setlocale value="en"/>

<fmt:setlocale value="en" scope="session"/>
```

<% taglib

%>

Taglib Format <fmt:/>

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>
```

- Afficher un message localisé
  - **<fmt:message />**
    - key (string) : clé dans le bundle (fichier properties)
    - bundle (localizationContext) : utiliser un bundle défini précédemment en utilisant setBundle`
    - var (string) : variable qui contiendra le message
    - scope (string) : scope où stocker la variable

```
<fmt:message key="message.title"/>

<fmt:message key="message.extended.title" bundle="extended"/>
```

<% taglib

%>

Taglib Format <fmt:/>

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>
```

- Afficher un message avec paramètre
  - **<fmt:param />**
    - value (object) : paramètre

monbundle_fr.properties

```
message.title=Bonjour les Français
message.home= Bonjour {0}, {1} message(s)
```

```
<fmt:message key="message.home">
   <fmt:param value="${user.name}"/>
   <fmt:param value="${user.newMessageCount}"/>
</fmt:message>
```

<% taglib

%>

Taglib Format <fmt:/>

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>
```

- Formatage
  - adapter les dates selon les fuseaux horaires
  - formater les dates/nombres selon des patterns et des types (date, time, devises, pourcentages)

<% taglib

%>

Taglib Format <fmt:/>

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>
```

- Définir un fuseau horaire
  - **<fmt:setTimeZone />**
    - value (object/timeZone) : fuseau
    - var (string) : variable où stocker le fuseau
    - scope (string) : scope où stocker la variable

```
<fmt:setTimeZone value="GMT+1 »>
```

- Utiliser un fuseau horaire
  - **<fmt:timeZone />**
    - value (object/timeZone) : fuseau à utiliser

<% taglib

%>

Taglib Format <fmt:/>

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>
```

- Formater une date (locale + fuseau)
  - **<fmt:dateFormat />**
    - value (date) : date à formater
    - type (string) : date / temps (date / time / both)
    - dateStyle (string) : style de la date (short, medium, long, full)
    - timeStyle (string) : style du temps (short, medium, long, full)
    - pattern (string) : pattern à utiliser pour formater la date
    - timeZone (string) : fuseau horaire à utiliser
    - var (string) : variable où stocker la date formatée
    - scope (string) : scope où stocker la variable

<% taglib

%>

Taglib Format <fmt:/>

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>
```

- Formater une date (locale + fuseau)
  - **<fmt:dateFormat />**
    - value (date) : date à formater
    - type (string) : date / temps (date / time / both)
    - dateStyle (string) : style de la date (short, medium, long, full)
    - timeStyle (string) : style du temps (short, medium, long, full)
    - pattern (string) : pattern à utiliser pour formater la date
    - timeZone (string) : fuseau horaire à utiliser
    - var (string) : variable où stocker la date formatée
    - scope (string) : scope où stocker la variable

<% taglib

%>

Taglib Format <fmt:/>

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>
```

- Exemple

```
<fmt:timeZone value="GMT+1">
   <fmt:formatDate value="${user.Birthday}" type="short"/>
</fmt:timeZone>
```

<% taglib

JSP Tag Library

%>

exemple de taglib

- Java Standard Tag Library (JSTL)
  - core (base) : http://java.sun.com/jsp/jstl/core
  - Format : http://java.sun.com/jsp/jstl/fmt
  - XML : http://java.sun.com/jsp/jstl/xml
  - SQL : http://java.sun.com/jsp/jstl/sql
  - Fonctions : http://java.sun.com/jsp/jstl/functions

<% taglib

Taglib XML <x:/>

%>

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/xml" prefix="x"%>
```

- Principales actions
  - accéder aux éléments d'un document XML avec XPATH
  - évaluer une expression XPATH (afficher)
  - Traitement conditionnel / itération sur le document XML (identique aux actions de la libraire core)
  - Parser un document xml (XML filter)
  - Transformer un document XML avec XSL

<% taglib

%>

Taglib XML <x:/>

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/xml" prefix="x"%>
```

- Charger un document XML
  - **<x:parse />**
    - doc (string/reader) : document à traiter
    - filter (XMLfilter) : filtre XML
    - var (string) : variable qui contiendra le XML
    - Scope (string) : scope où sera stocké la variable
    - varDom : idem var mais sous forme d'objet DOM
    - scopeDom (string) : scope où stocker la variable

```
<x:parse doc="${cv.xml}" var="parsedCv"/>

<x:parse var="parsedCv"/>
    <cv:cv><title>CV</title></cv:cv>
<x:parse/>
```

<% taglib

%>

Taglib XML <x:/>

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/xml" prefix="x"%>
```

- Evaluer une expression XPATH
  - **<x:out />**
    - select (string) : expression XPATH
    - escapeXML (boolean) : echappement XML

```
<x:out select= "{$parsedCV/title}"/>
```

<% taglib

%>

Taglib XML <x:/>

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/xml" prefix="x"%>
```

- Transformation XSL
  - **<x:transform />**
    - doc (object) : document XML à transformer
    - xslt (object) : feuille de transformation
    - var (string) : variable de sortie
    - scope (string) : scope où sera stocké la variable
    - result (result) : capture du résultat de la transformation

```
<x:transform doc="${parsedCV}" xslt="${stylesheet}" />
```

<% taglib

Taglib Functions <fn:/>

%>

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>
```

- Principales actions
  - actions des base
    - gestion des chaines de caractères

<% taglib

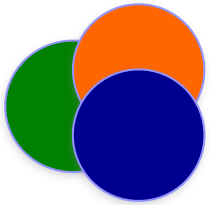
Taglib Functions <fn:/>

%>

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>
```

- Principales actions
  - opérations booléennes sur les chaines
    - string => boolean
    - fn:contains(), f:containsIgnoreCase(), fn:endsWith()
  - opérations sur les chaines
    - string => string
    - fn:escapeXml(), fn:indexOf(), fn:joint(), fn:length(), contains(), f:containsIgnoreCase(), fn:endsWith(), fn:startsWith(), fn:replace(), fn:split(), fn:subString(), fn:subStringAfter(), fn:subStringBefore(), fn:toLowerCase(), fn:toUpperCase(), fn:trim()

Framework Spring

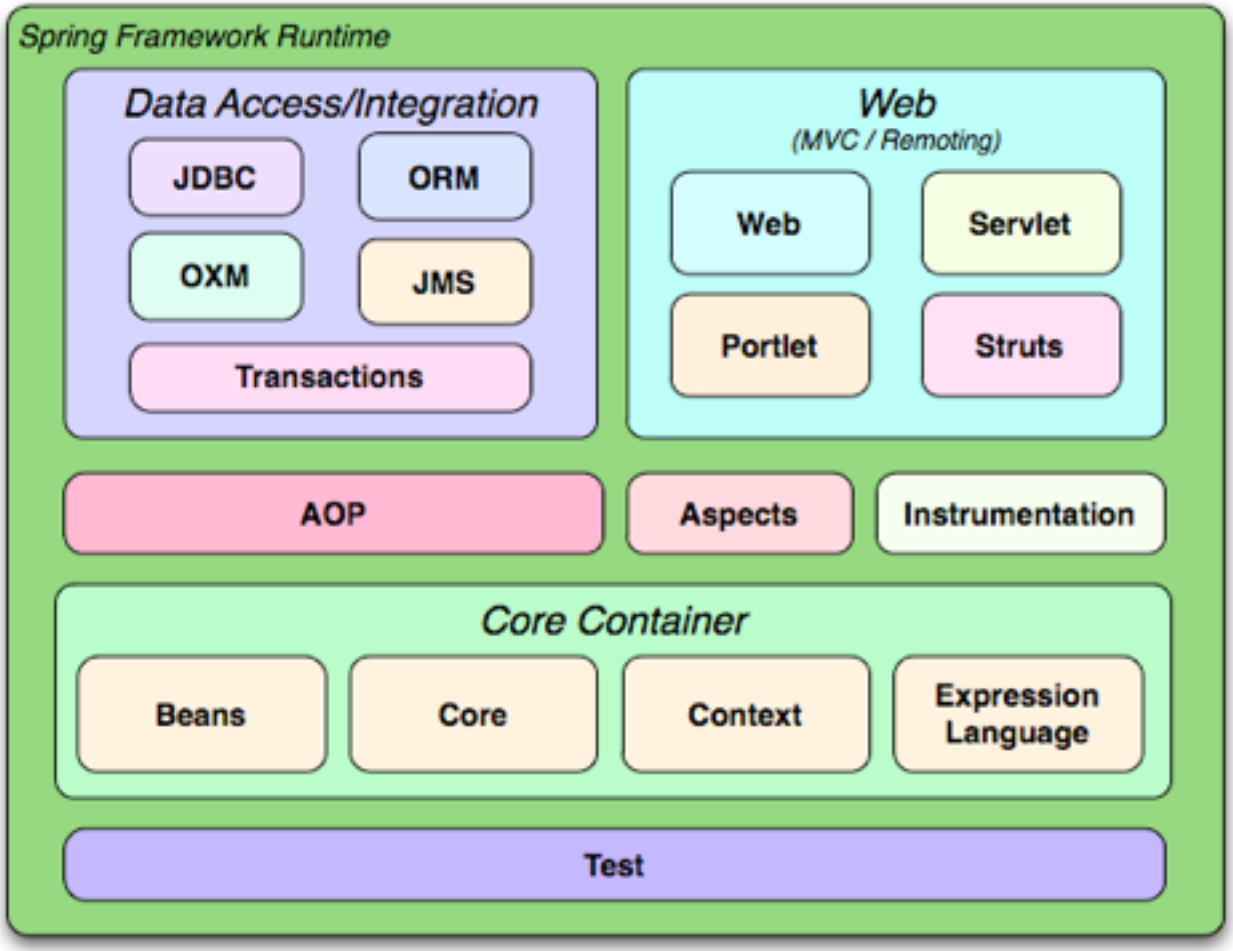


Framework Spring

- Framework d'aide au développement d'applications Java
  - POJO (Plain Old Java Object)
  - Java EE
  - Infrastructure légère / JEE (peu de dépendance aux interfaces)

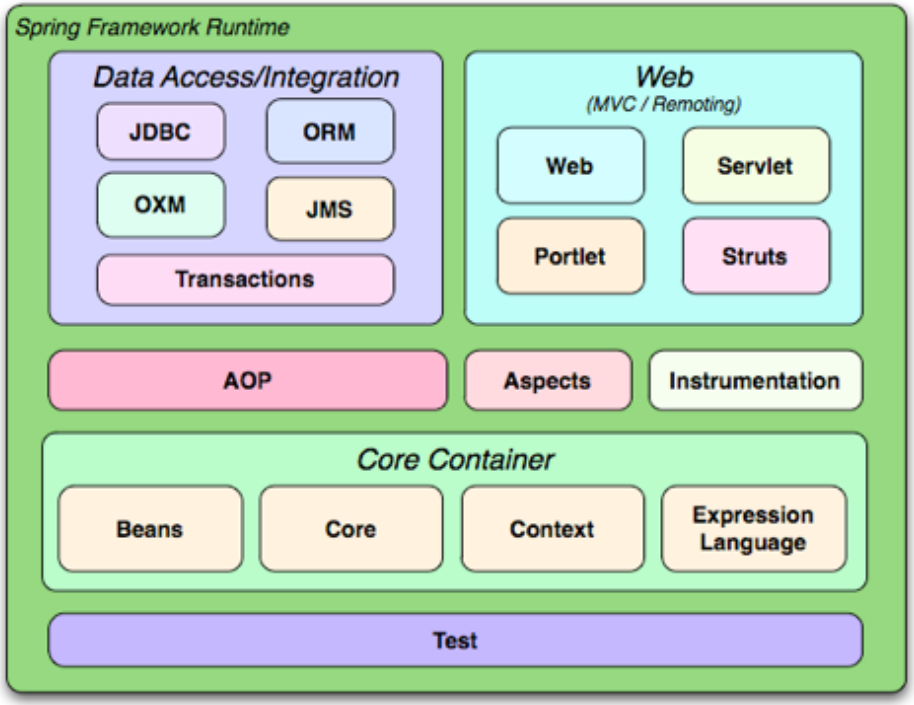
- Objectifs :
  - Apporter une couche d'**abstraction** aux JSR / implémentations
  - Gérer les **dépendances** entres les objets de l'application (**injection**)
  - Organiser et mettre en **cohérences** les objets de l'application (composition)
  - Simplifier l'utilisation des **design pattern** en proposant une implémentation
  - **Accélérer** le développement

Modules Spring (20)



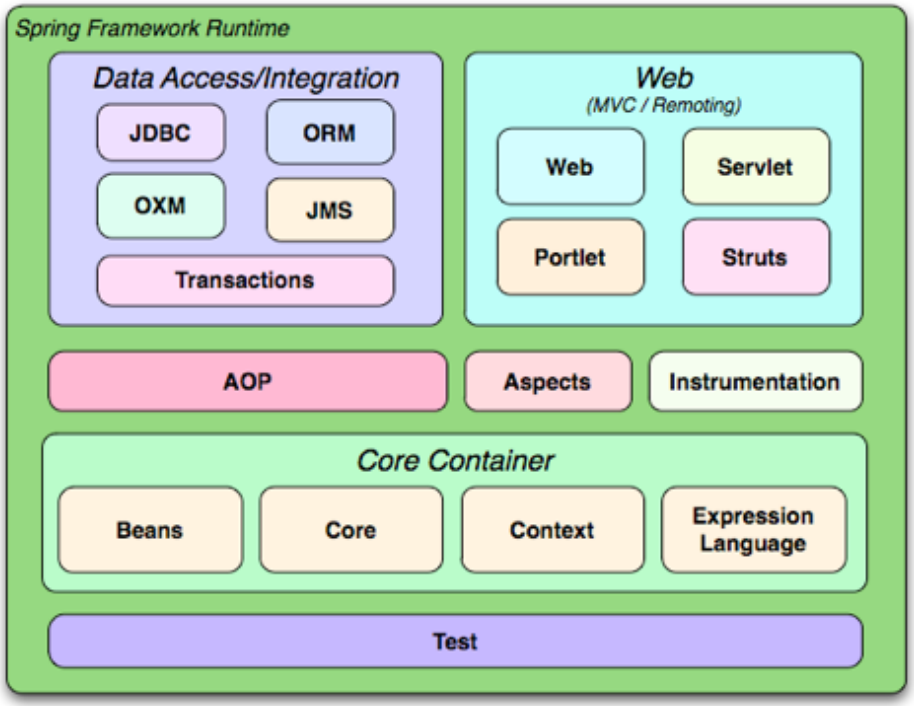
Modules Spring (20)

- Core
  - Core
    - Injection de dépendances
  - Beans
    - Factory
  - Context
    - Internationalisation
    - Bundle
    - EJB
    - Servlet context
  - Expression Language
    - Interpréteur Java
    - Utilisation des EL dans XML



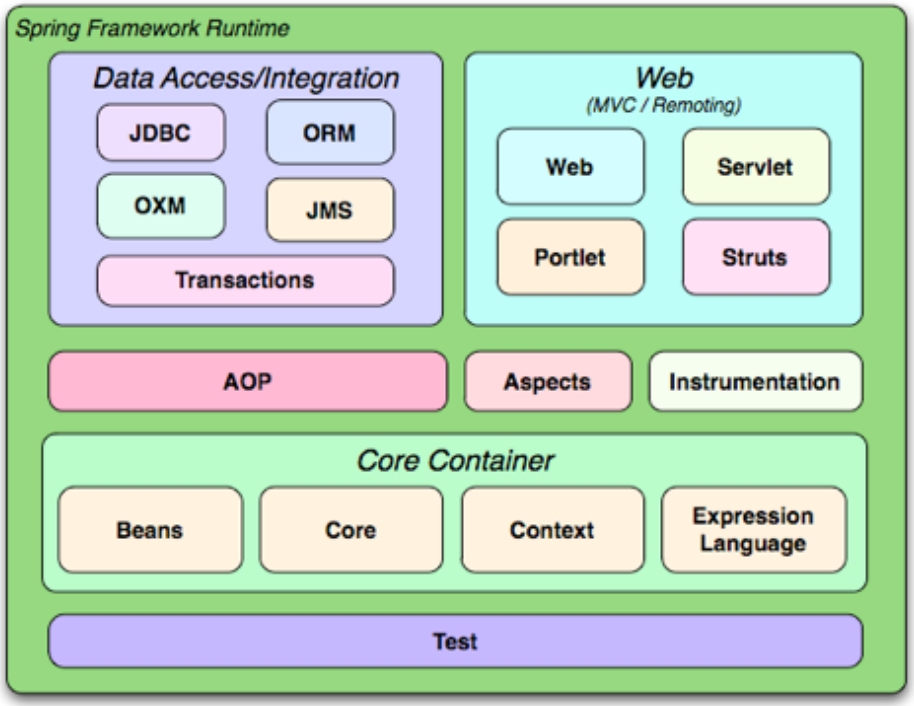
Modules Spring (20)

- Data Integration
  - JDBC
    - abstraction
  - **O**bject-**R**elationnal **M**apping
    - JPA
    - Hibernate
    - iBatis
  - **O**bject / **X**ML **M**apping
    - parsing
    - implémentations
  - **J**ava **M**essaging **S**ervice
  - Transactions
    - POJO

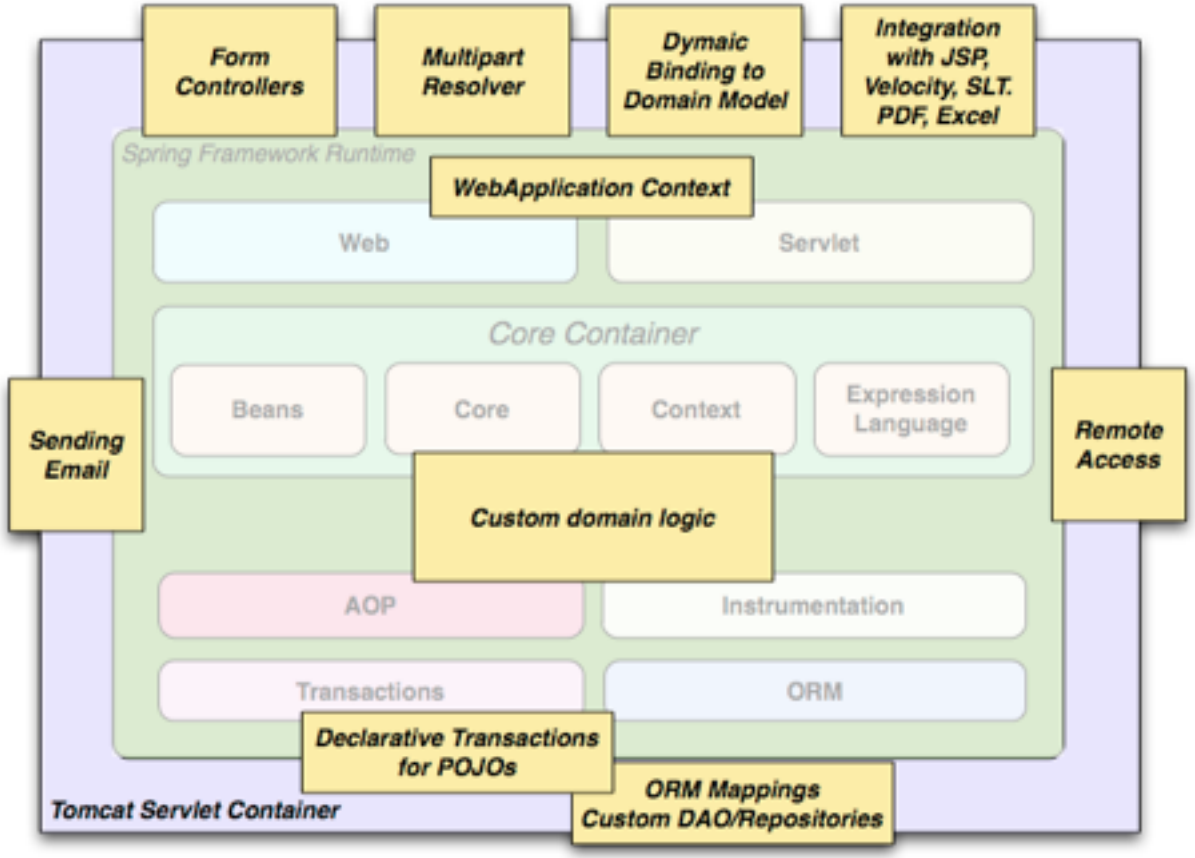


Modules Spring (20)

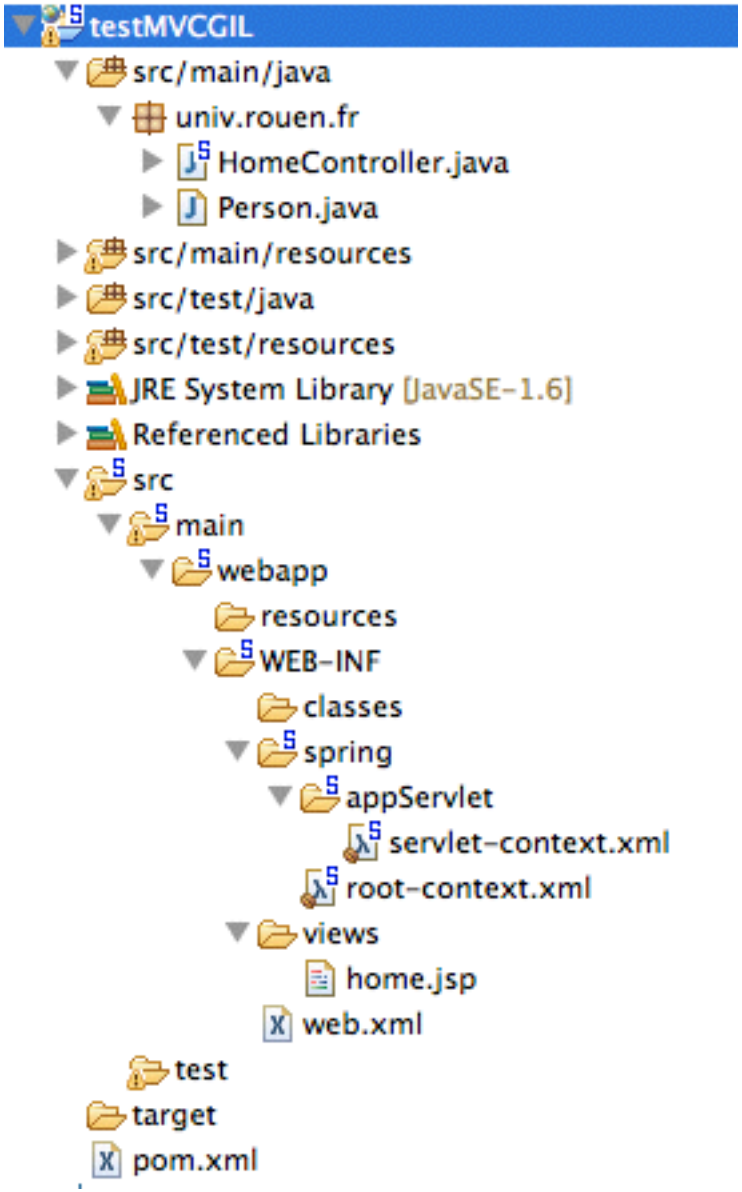
- Web
  - Web
    - Servlet-listener (init)
    - Web Binding
    - Web Validator
    - Construction de formulaire
  - Servlet
    - MVC Servlet
  - Portlet
    - MVC Portlet
  - Struts
    - Flow-control (mapping)



Modules Spring (20)



Projet Spring



web.xml

```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>/WEB-INF/spring/root-context.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>

    <!-- Processes application requests -->
    <servlet>
        <servlet-name>appServlet</servlet-name>
        <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
        <init-param>
            <param-name>contextConfigLocation</param-name>
            <param-value>/WEB-INF/spring/appServlet/servlet-context.xml</param-value>
        </init-param>
        <load-on-startup>1</load-on-startup>
    </servlet>

    <servlet-mapping>
        <servlet-name>appServlet</servlet-name>
        <url-pattern>/</url-pattern>
    </servlet-mapping>

</web-app>
```

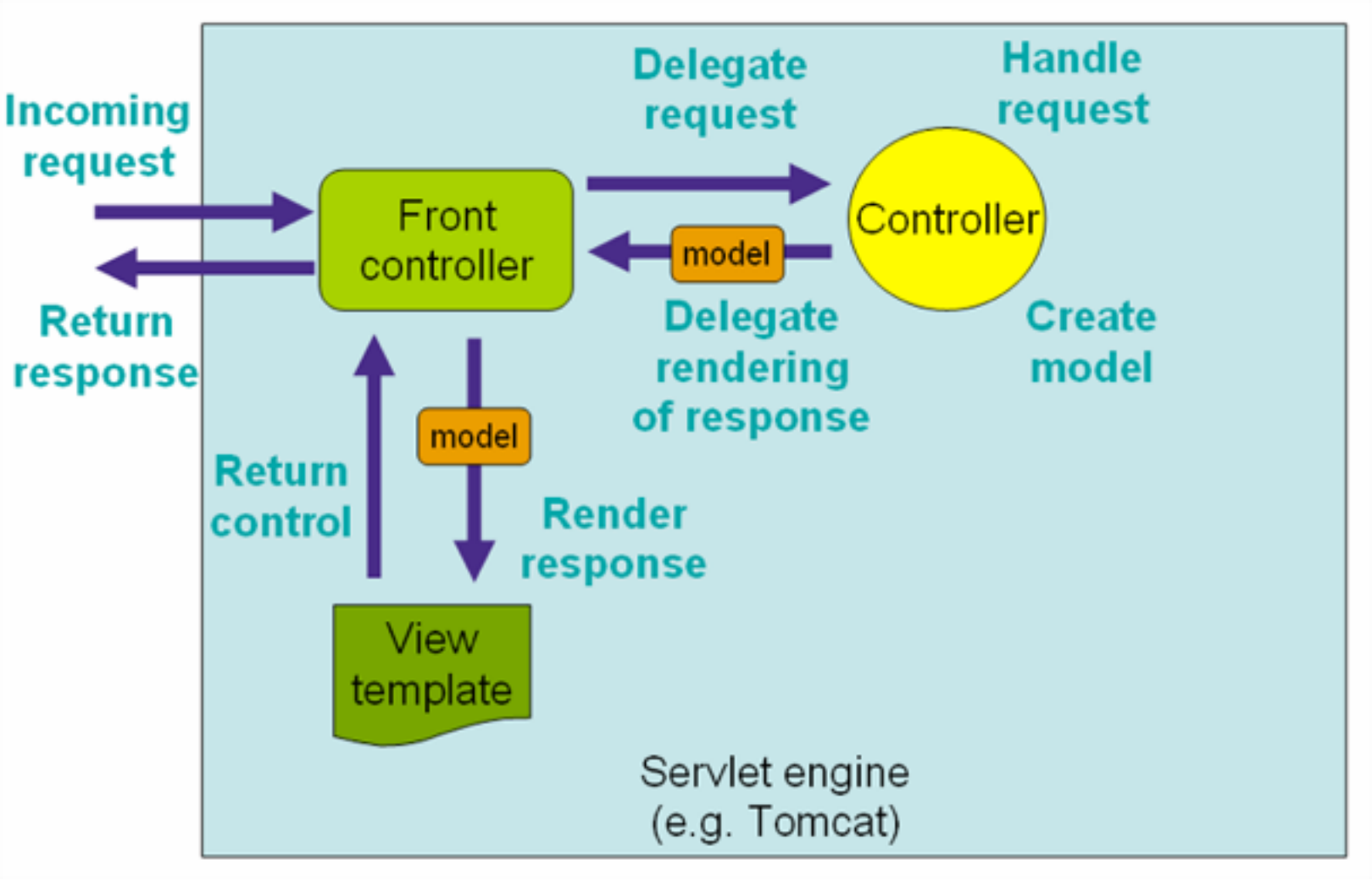
servlet-context.xml

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

    <!-- DispatcherServlet Context: defines this servlet's request-processing infrastructure -->

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

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

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

    <context:component-scan base-package="univ.rouen.fr" />


</beans:beans>
```

Beans (déclaration)

```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-3.0.xsd">

  <bean id="monBean1" class="univ-rouen.test.bean">
    <property name="nom" value="Edison"/>
    <property name="prenom" value="Thomas"/>
  </bean>

</beans>
```

MVC (actions based)



Model (`org.springframework.ui.Model`)

- Extension de Map
  - Attributs : clé / valeur
  - Gestion automatique des attributs de session par Spring
  - Construit dans le contrôleur et passé à la vue
- Extension de Map
  - Annotations Java

Controller

```java
/**
 * Handles requests for the application home page.
 */
@Controller
// model attributes managed by the spring controller
@SessionAttributes({ "login", "nb_messages" })
public class HomeController {
	/**
	 * My person bean (injected)
	 */
	@Autowired
	Person personDefault;

	/**
	 * A Spring service to manage person (business tier)
	 */
	@Autowired
	ServicePerson service;

	/**
	 * Simply selects the home view to render by returning its name.
	 */
	@RequestMapping(value = "/", method = RequestMethod.GET)
	public String home(Person person, Model model) {

		// calling the business service
		int nbMess = service.getNbMessage(person.getNom()+person.getPrenom());

		// adding attributes to the model
		model.addAttribute("nb_messages", nbMess);
		model.addAttribute("firstName", person.getPrenom());
		model.addAttribute("lastName", person.getNom());
		// return the view name
		return "home";
	}
}
```

Vue

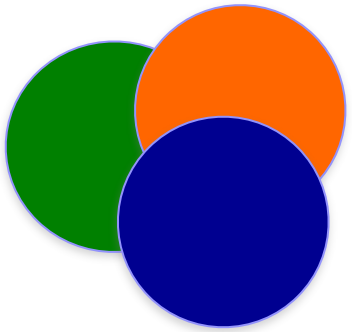
- Validation des formulaires
  - WebDataBinder
  - Validator

```
<%@ taglib prefix="form"
           uri="http://www.springframework.org/tags/form" %>

<form:form>
      <form:errors path="*" cssClass="errorBox" />
      <table>
          <tr>
              <td>First Name:</td>
              <td><form:input path="firstName" /></td>
              <td><form:errors path="firstName" /></td>
          </tr>
          <tr>
              <td>Last Name:</td>
              <td><form:input path="lastName" /></td>
              <td><form:errors path="lastName"  /></td>
          </tr>
          <tr>
              <td colspan="3">
                  <input type="submit" value="Save Changes" />
              </td>
          </tr>
      </table>
  </form:form>
```

Framework JSF

=

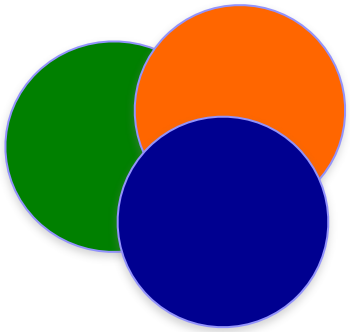
+



Framework JSF

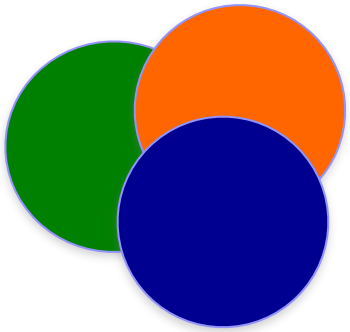
- Managed Beans
  - Annotations Java

- Vue
  - TagLib jsp
- Navigation
  - fichier xml règles (faces-config.xml)
  - attribut « action » d'un bouton



Framework JSF

- testJSF
  - src/main/java
    - univ.rouen.fr
      - Biography.java
      - Fact.java
  - Referenced Libraries
  - JRE System Library [JavaSE-1.6]
  - build
  - src
    - main
      - webapp
        - WEB-INF
          - web.xml
        - biography.xhtml
        - hello.xhtml
  - target
  - pom.xml



Framework JSF

```java
import javax.faces.bean.ManagedBean;
import javax.faces.bean.SessionScoped;

/**
 * Session Bean implementation class Biography
 */
@ManagedBean
@SessionScoped
public class Biography {
    private String nom, prenom;
    private Fact[] faitsNotables;

    public Biography() {}

    public String getPrenom() {
        return prenom;
    }
    public void setPrenom(String prenom) {
        this.prenom = prenom;
    }
    public String getNom() {
        return nom;
    }
    public Fact[] getFaitsNotables() {
        return faitsNotables;
    }
    public void setFaitsNotables(Fact[] faitsNotables) {
        this.faitsNotables = faitsNotables;
    }
    public void setNom(String nom) {
        this.nom = nom;
        if (nom.equals("Edison")) {
            setPrenom("Thomas");
            Fact[] faits = new Fact[2];
            Fact f = new Fact();
            f.setAnnee(1930);
            f.setTitre("Invention de l'électricité");
            faits[0] = f;
            Fact f2 = new Fact();
            f2.setAnnee(1935);
            f2.setTitre("Invention du cinéma");
            faits[1] = f2;
            setFaitsNotables(faits);
        }
    }
}
```

Framework JSF


```
<?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:f="http://java.sun.com/jsf/core" xmlns:h="http://java.sun.com/jsf/html">
    <h:head>
        <title>JSF 2.0 Biography</title>
    </h:head>
    <h:body>
        <h:form>
           <h:inputText value="#{biography.nom}"></h:inputText>
           <h:commandButton value="Search this person" action="biography"></h:commandButton>
        </h:form>
    </h:body>
</html>
```

```
<?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:h="http://java.sun.com/jsf/html" xmlns:f="http://java.sun.com/jsf/core">
<h:head>
        <title>JSF 2.0 Biography</title>
</h:head>
<h:body bgcolor="white">
        <h3>JSF 2.0 Example - biography.xhtml</h3>
        <h4>Biography of #{biography.nom} #{biography.prenom}</h4>
        <h:dataTable value="#{biography.faitsNotables}" var="item" cellspacing="0" cellpadding="1" border="1">
                <h:column>
                        <f:facet name="header">Year</f:facet>
                        <h:outputText value="#{item.annee}" />
                </h:column>
                <h:column>
                        <f:facet name="header">Fact</f:facet>
                        <h:outputText value="#{item.titre}" />
                </h:column>
        </h:dataTable>
</h:body>
</html>
```

Framework JSF


```
<?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:f="http://java.sun.com/jsf/core" xmlns:h="http://java.sun.com/jsf/html">
    <h:head>
        <title>JSF 2.0 Biography</title>
    </h:head>
    <h:body>
          <h:form>
             <h:inputText value="#{biography.nom}"></h:inputText>
             <h:commandButton value="Search this person" action="biography"></h:commandButton>
          </h:form>
    </h:body>
</html>
```


```
<?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:h="http://java.sun.com/jsf/html" xmlns:f="http://java.sun.com/jsf/core">
<h:head>
          <title>JSF 2.0 Biography</title>
</h:head>
<h:body bgcolor="white">
          <h3>JSF 2.0 Example - biography.xhtml</h3>
          <h4>Biography of #{biography.nom} #{biography.prenom}</h4>
          <h:dataTable value="#{biography.faitsNotables}" var="item" cellspacing="0" cellpadding="1" border="1">
                    <h:column>
                              <f:facet name="header">Year</f:facet>
                              <h:outputText value="#{item.annee}" />
                    </h:column>
                    <h:column>
                              <f:facet name="header">Fact</f:facet>
                              <h:outputText value="#{item.titre}" />
                    </h:column>
          </h:dataTable>
</h:body>
</html>
```

JSF 2.0 Example - biography.xhtml

Biography of Edison Thomas

Année	Fait
1930	Invention de l'électricité
1935	Invention du cinéma

Framework JSF


```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xmlns="http://java.sun.com/xml/ns/javaee"
        xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
          xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
          http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
          id="WebApp_ID" version="2.5">

          <display-name>JavaServerFaces</display-name>

          <!-- Change to "Production" when you are ready to deploy -->
          <context-param>
                    <param-name>javax.faces.PROJECT_STAGE</param-name>
                    <param-value>Development</param-value>
          </context-param>

          <!-- Welcome page -->
          <welcome-file-list>
                    <welcome-file>faces/hello.xhtml</welcome-file>
          </welcome-file-list>

          <!-- JSF mapping -->
          <servlet>
                    <servlet-name>Faces Servlet</servlet-name>
                    <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
                    <load-on-startup>1</load-on-startup>
          </servlet>

          <!-- Map these files with JSF -->
          <servlet-mapping>
                    <servlet-name>Faces Servlet</servlet-name>
                    <url-pattern>/faces/*</url-pattern>
          </servlet-mapping>
          <servlet-mapping>
                    <servlet-name>Faces Servlet</servlet-name>
                    <url-pattern>*.jsf</url-pattern>
          </servlet-mapping>
          <servlet-mapping>
                    <servlet-name>Faces Servlet</servlet-name>
                    <url-pattern>*.faces</url-pattern>
          </servlet-mapping>
          <servlet-mapping>
                    <servlet-name>Faces Servlet</servlet-name>
                    <url-pattern>*.xhtml</url-pattern>
          </servlet-mapping>

</web-app>
```

Framework JSF


```
<faces-config version="1.2" 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-facesconfig_1_2.xsd">
	<application>
		<el-resolver>jsf.util.JsfCrudELResolver</el-resolver>
	</application>
	<navigation-rule>
		<navigation-case>
			<from-outcome>welcome</from-outcome>
			<to-view-id>/hello.xhtml</to-view-id>
		</navigation-case>
	</navigation-rule>
	<navigation-rule>
		<navigation-case>
			<from-outcome>biography</from-outcome>
			<to-view-id>biography.xhtml</to-view-id>
		</navigation-case>
	</navigation-rule>
	<managed-bean>
		<managed-bean-name>bio</managed-bean-name>
		<managed-bean-class>univ.rouen.fr.Biography</managed-bean-class>
		<managed-bean-scope>session</managed-bean-scope>
	</managed-bean>

	<converter>
		<converter-for-class>jpa.entities.Biography</converter-for-class>
		<converter-class>univ.rouen.fr.BioConverter</converter-class>
	</converter>

</faces-config>
```

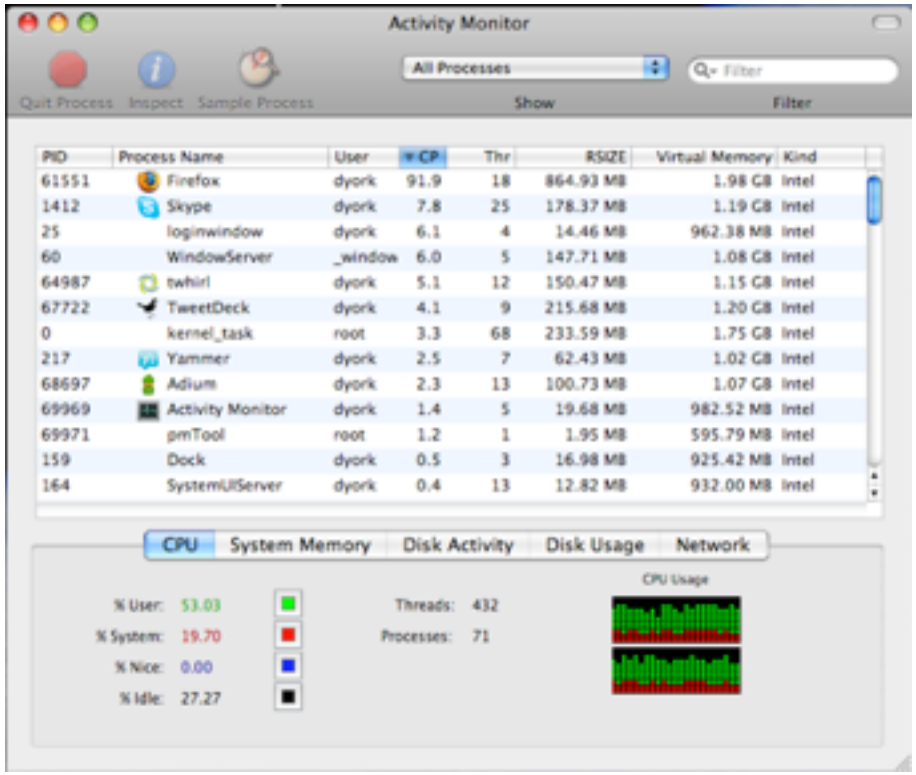
Technologie AJAX

- Ensemble de technologies (2005)
  - JavaScript, XML, CSS, DOM, XMLHttpRequest …
- Objectifs :
  - Construire des applications Web **dynamiques et interactives**
  - **Optimiser les interactions** dans les applications client / serveur
  - **Limiter la quantité** de données échangées entre le client / serveur

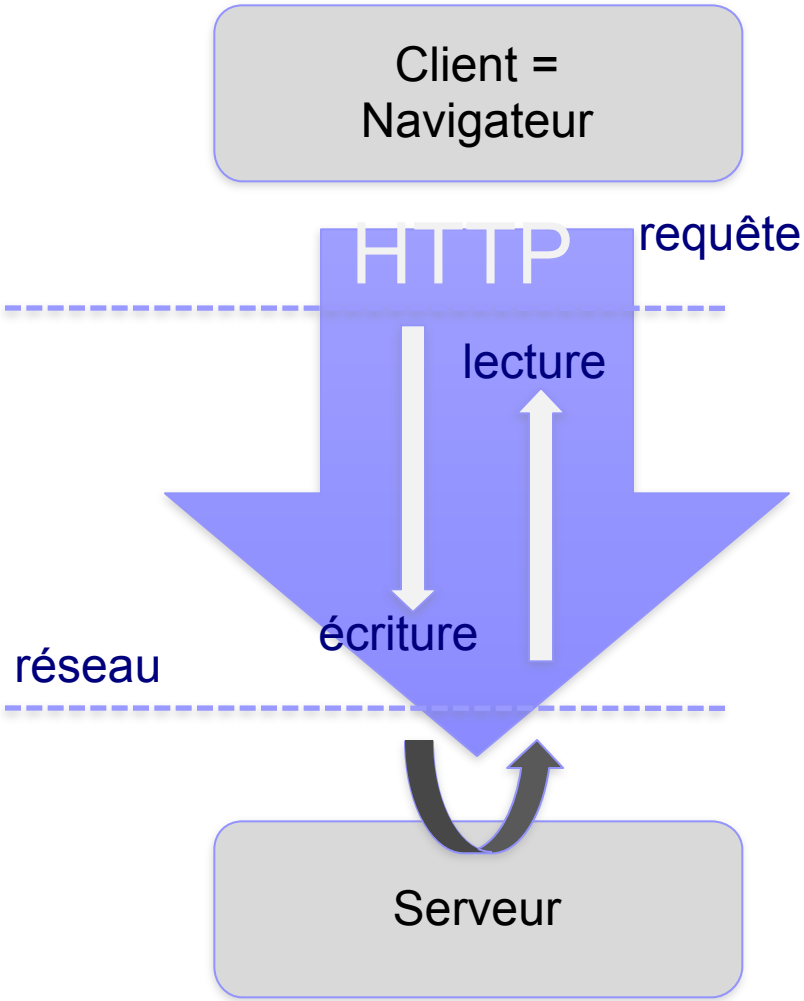
➢ **Préserver les ressources serveur (réseau, processeur, etc.) en exploitant les ressources client pour développer des applications plus réactives**

Technologie AJAX

- **Préserver les ressources serveur (réseau, processeur, etc.) en exploitant les ressources client pour développer des applications plus réactives**



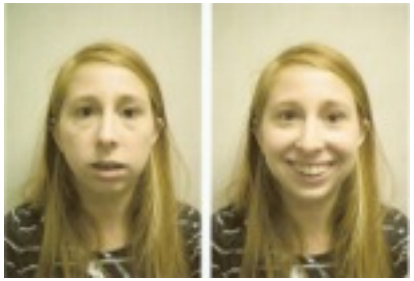
HTTP, client et serveur

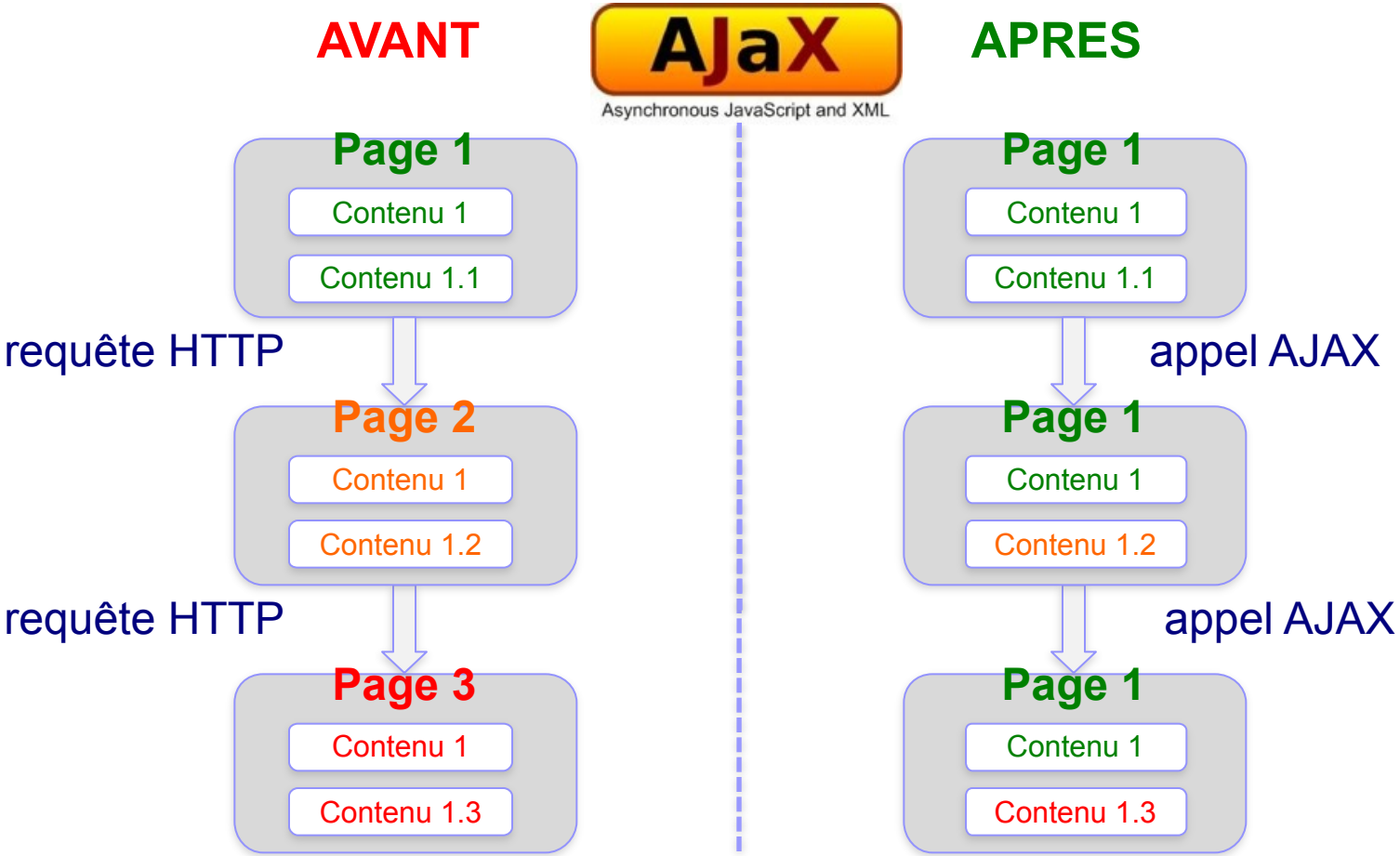


unidirectionnel
client -> serveur

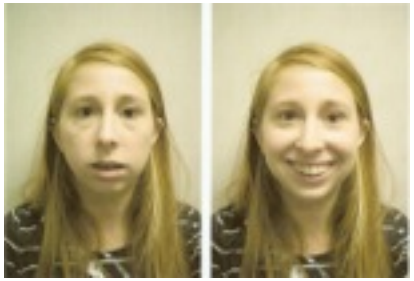
mise à jour de la vue = demande du client

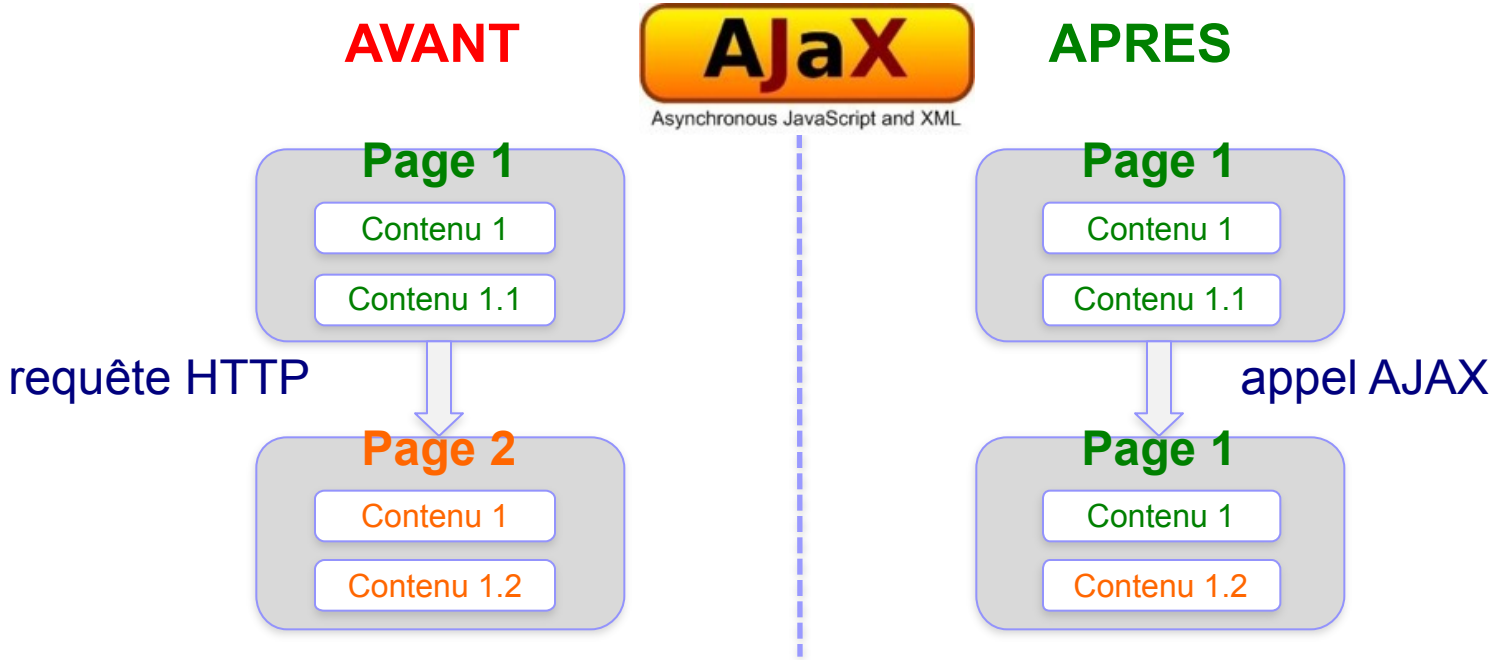
AJAX = Asynchronous





AJAX = Asynchronous





- conditions / cas d'usage :
  - contenu 1 identique dans toutes les pages
  - le passage du contenu 1.1 à 1.n n'empêche pas la navigation
  - contenus 1.x ne seront pas obligatoirement vus par l'utilisateur

Asynchrone…

exemple

= AJAX

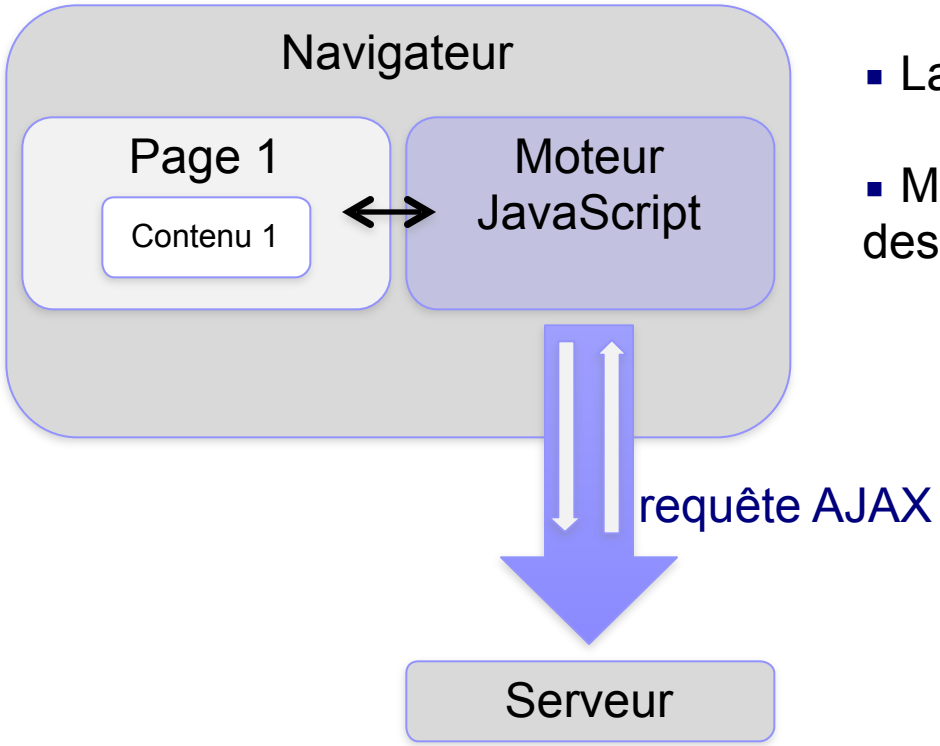
(souvent …)

… mais la page n’est pas gelée

Asynchrone…

exemple

AJAX = JavaScript



- Lance les requêtes AJAX
- Met à jour le contenu de la page à partir des réponses

JavaScript : Implémentation

- Objet **`XMLHttpRequest`** *(Recommandation W3C en cours)*
  - Lit les donnés de façon asynchrone sur un serveur
  - URL du serveur
  - paramètres
  - méthode HTTP (post / get)
  - header HTTP (encoding, cookie, etc.)

- Nécessite un navigateur supportant la méthode…

JavaScript : Implémentation

- Objet **`XMLHttpRequest`** : méthodes
  - `open()` : fixe la requête
  - `send()` : envoi la requête au serveur
  - `abord()` : arrête la requête
  - `getAllResponseHeader()` : récupère les entête des réponses
  - `getResponseHeader()` : récupère l'entête de la réponse
  - `getRequestHeader()` : pour fixer l'entête de la requête
  - `onreadystatechange()` : événement envoyé lors des changements d'état
  - `readystate()` : statut de la requête
  - `responseText()` : contenu texte de la réponse
  - `responseXML()` : contenu XML de la réponse
  - `status()` : statut HTTP
  - `statusText()` : version texte du statut HTTP

- Objet pas si simple à utiliser nativement …

JavaScript : Implémentation

AJaX
Asynchronous JavaScript and XML

(simplifiée)

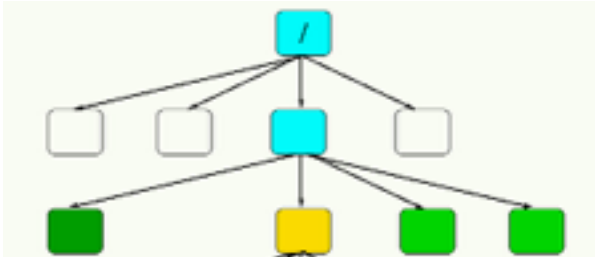
- Objet **`XMLHttpRequest :`**

```
<html>
    <body>
        <script type="text/javascript">

            // création de l'objet
            xhr = new XMLHttpRequest();

            // appel asynchrone au serveur
            xhr.open('GET', 'http://monserveur.fr/page.jsp', true);

            // parametres de l'appel
            xhr.call('chapitre=1&paragraphe=4');

            // récupération de la réponse
            data = xhr.responseXML;

        </script>
    </body>
</html>
```

AJAX : XML

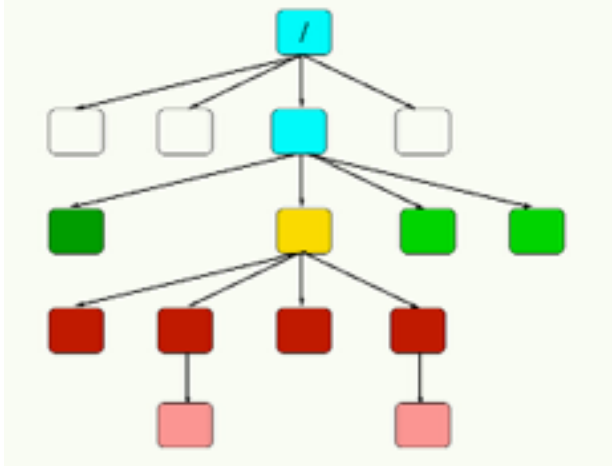
Asynchronous JavaScript and XML

Arbre DOM de la page HTML



Réponse XML de la requête AJAX

Mise à jour de l'arbre DOM en JavaScript



JavaScript : Implémentation

- Mise à jour de l'arbre DOM depuis **`XMLHttpRequest :`**

```
<html>
    <body>
        <script type="text/javascript">

            // création de l'objet
            xhr = new XMLHttpRequest();

            // appel asynchrone au serveur
            xhr.open('GET', 'http://monserveur.fr/page.jsp', true);

            // parametres de l'appel
            xhr.call('chapitre=1&paragraphe=4');

            // récupération de la réponse
            data = xhr.responseXML;

            // mise à jour de l'arbre DOM de la page
            document.getElementById('monElement').innerHTML = data;

        </script>
    </body>
</html>
```

JavaScript : Implémentation

- Mise à jour de l'arbre DOM depuis **`XMLHttpRequest :`**

```
// mise à jour de l'arbre DOM de la page
document.getElementById('monElement').innerHTML = data;
```

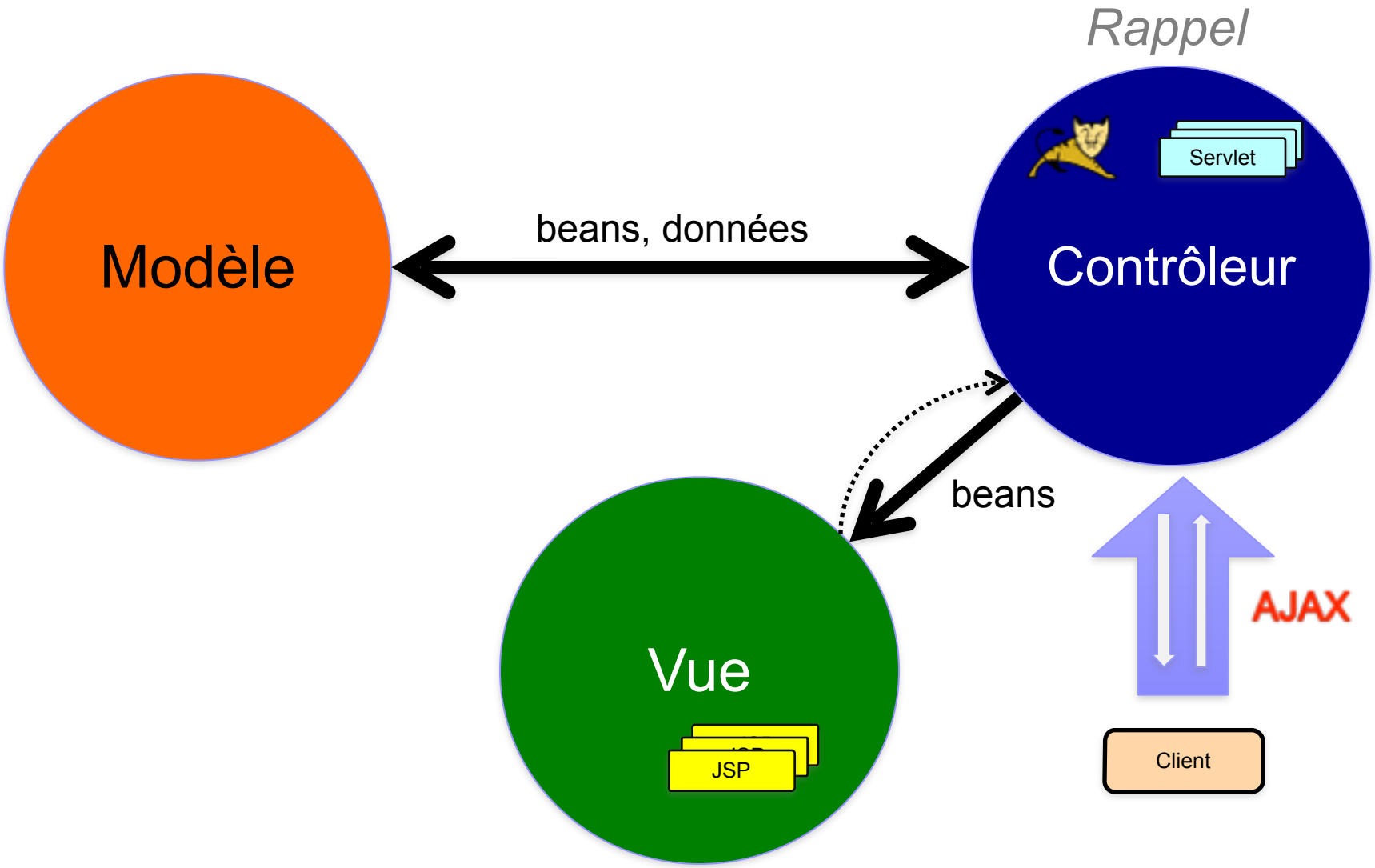
```
<html>
   <body>
      <h1>My First Heading</h1>
      <p>My first paragraph. </p>
      <div id="monElement">

      </div>
   </body>
</html>
```

```
<html>
   <body>
      <h1>My First Heading</h1>
      <p>My first paragraph. </p>
      <div id="monElement">

        Content dynamically loaded by AJAX !

      </div>
   </body>
</html>
```

À VOUS DE JOUER!

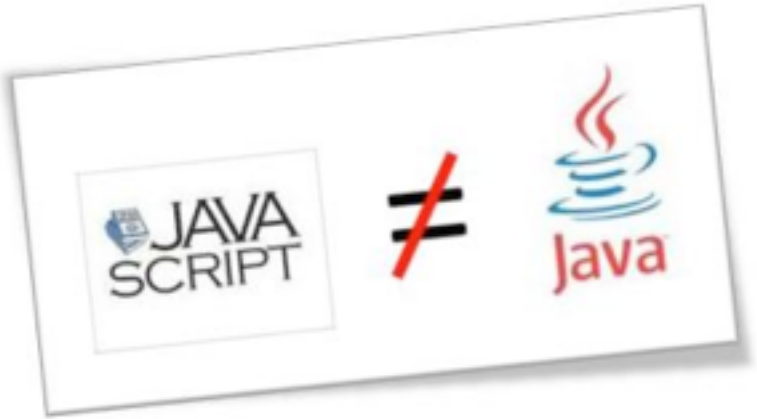
```
<%@ page language="java" contentType="text/html;" pageEncoding="utf-8"%>
<%
String data = request.getParameter("data");
String selected = request.getParameter("selected");
if (selected == null) {
%>
<html>
    <head><title>Exercice AJAX</title></head>
    <body>
    <div id="result"></div>
    <form method="GET" action="exo.jsp">
        <input type="text" maxlength="50" name="data" id="data"/>
        <select name="selected" id="selected" onchange="getInfosToDisplay();">
            <option value="-1">select</option>
            <option value="0">I</option>
            <option value="1">F</option>
        </select>
    </form>
    <script type="text/javascript">
    function getInfosToDisplay() {
        url = "exo.jsp?data=" + document.getElementById("data").value +
         "&selected=" + document.getElementById("selected").value;
        methodToCall = "useTxtInfo";
        loadXMLDoc(url,methodToCall);
    }
    function loadXMLDoc(url,methodToCall) {
      if (window.XMLHttpRequest) {
        objReq= new XMLHttpRequest();
        objReq.onreadystatechange = eval(methodToCall);
        objReq.open("GET", url, true);
        objReq.send(null);
      } else if (window.ActiveXObject) {
        objReq = new ActiveXObject("Microsoft.XMLHTTP");
        if ( objReq) {
          objReq.onreadystatechange = eval(methodToCall);
          objReq.open("GET", url, true);
          objReq.send();}}
    }
    function useTxtInfo() {
      if ( objReq.readyState == 4) {
        if (objReq.status == 200) {
           serverAnswer = objReq.responseText;
           document.getElementById("result").innerHTML = serverAnswer;
    }}}
    </script>
</body>
</html>
<%
} else {
    if (selected.equals("0")) {
        try{Integer.valueOf(data);response.getWriter().write("why not");
        } catch (Exception e) {response.getWriter().write("for sure not");}
    }else if (selected.equals("1")) {
        try{Float.valueOf(data);response.getWriter().write("why not");
        }  catch (Exception e) {response.getWriter().write("for sure not");}}
}
%>
```

Java EE & MVC + AJAX



JavaScript n'est pas java…

- Java et Javascript sont des langages orientés objet mais …
- Un objet Javascript = tableau …
- Bugs



JavaScript : help !

- Debugger dans les navigateurs
  - ✓ Firebug
    - suivi des erreurs
    - mise à jour des scripts
    - visualisation de l'arbre DOM
    - visualisation des échanges HTTP

- Objet Java -> objet JavaScript :
  - ✓ JSON (JavaScript Object Notation)

JSON vs XML

- JSON avantages :
  - vitesse de traitement
  - simplicité de mise en œuvre
  - pas besoin de parser le XML pour extraire les informations
  - reconnu nativement par JavaScript

- XML avantages :
  - extensible
  - reconnu par (presque) tous les langage de programmation
  - plus facile à lire et à écrire …

Exemple


```
var courses = {
  "fruits": [
    { "kiwis": 3,
      "mangues": 4,
      "pommes": null
    },
    { "panier": true },
  ],
  "legumes":
    { "patates": "amandine",
      "figues": "de barbarie",
      "poireaux": false
    }
};
```

```
<root>
   <fruits>
      <item>
         <kiwis>3</kiwis>
         <mangues>4</mangues>
         <pommes></pommes>
      </item>
      <item>
         <panier>true</panier>
      </item>
   </fruits>
   <legumes>
      <patates>amandine</patates>
      <figues>de barbarie</figues>
      <poireaux>false</poireaux>
   </legumes>
</root>
```

JavaScript : help !

(suite)

- Framework JavaScript
  - jQuery
  - EXT
  - Prototype
  - DOJO
  - DWR
  - …

jQuery
write less, do more.

prototype

Ext JS

DWR

dojo

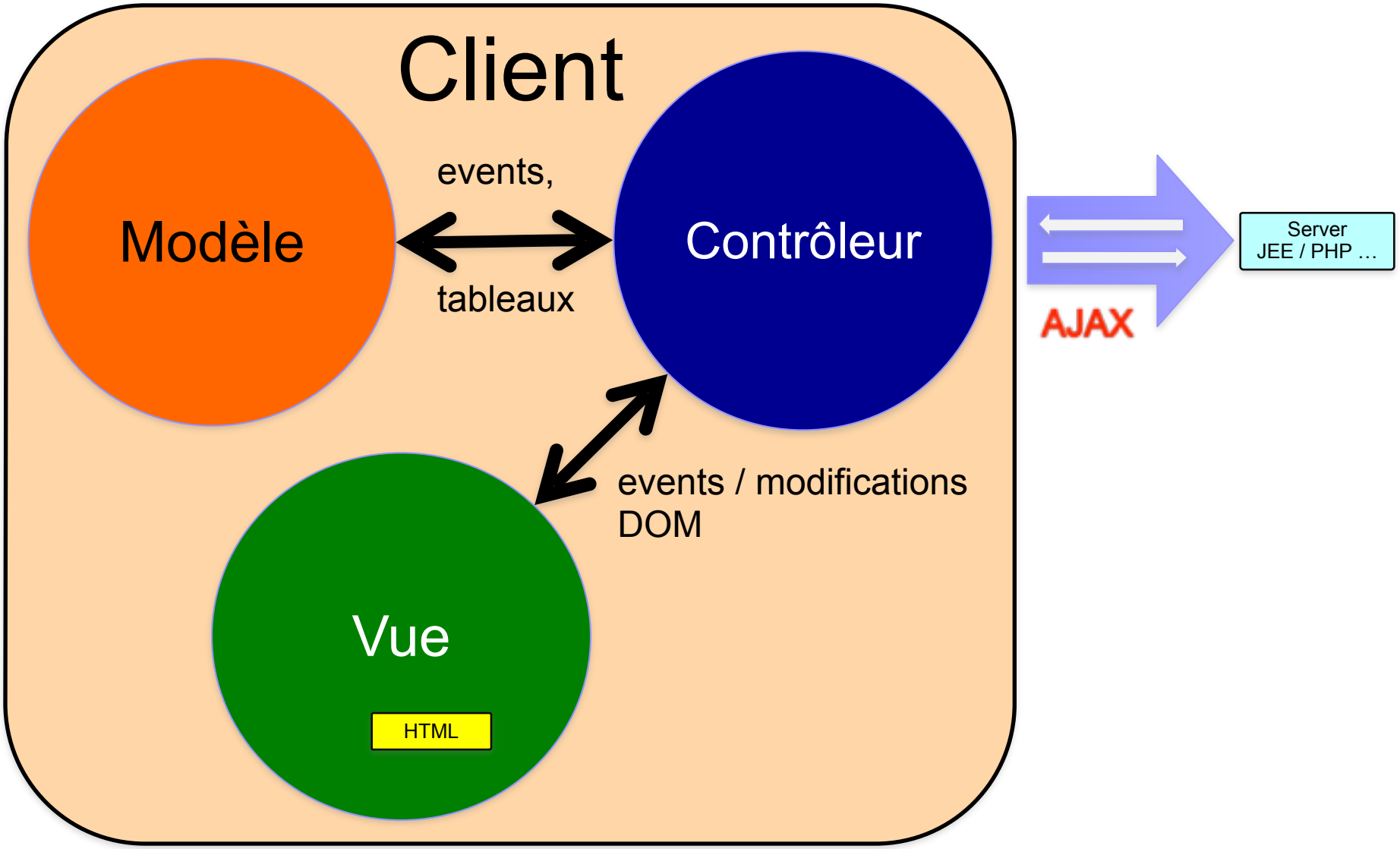
JavaScript & MVC

en utilisant AJAX…

avec ou sans Java…

- MVC en JavaScripts
  - fonctionnent en utilisant AJAX pour la récupération /sauvegarde des données
  - les trois composantes Vue, Modèle et Contrôleur sont coté client.
- Framework
  - Backbones, Ember.js, Spine.js, Google Web Toolkit (GWT), …

JavaScript, AJAX & MVC



Au menu

- Taglibs (suite)
- Spring
- AJAX – présentation
- AJAX & MVC : Java vs JavaScript
- AJAX – Présentation du framework jQuery

JavaScript : jQuery

- Manipulation du DOM :
  - Recherche d'éléments DOM
  - Ajout de nouveaux éléments
  - Suppression d'éléments
  - Modification des attributs des éléments
  - Modification du style / position des éléments
  - Gestion des événements
  - AJAX
  - Effets visuels
  - Nombreux plugins…

JavaScript : jQuery

- Quelques méthodes jQuery pour l'AJAX :
  - `jQuery.load()` : appel AJAX + remplacement DOM
  - `jQuery.get()` : charge des donnés texte/XML
  - `jQuery.post()` : charge des donnés en POST
  - `jQuery.getJSON()` : charge des données JSON

- Quelques méthodes jQuery pour la sélection :
  - `jQuery.('#id')` : éléments possédant cet id
  - `jQuery.('.class')` : éléments possédant cette classe

JavaScript : jQuery

- Exemple

```
<html>
    <head>
        <script src="http://code.jquery.com/jquery-latest.js"></script>
    </head>
    <body>
        <h1>My First Heading</h1>
        <p>My first paragraph. </p>
        <div id="monElement">

        </div>

        <script>
            jQuery("#monElement").load("http://monserver.fr/page.jsp");
        </script>
    </body>
</html>
```

AJAX : all-in-one