

SIL4

Programmation Web en Java

Francis Brunet-Manquat – MIAM (bureau 112)

Hervé Blanchon – SIMO (bureau 105)

Basé sur le cours de Leila Kefi-Khelif et D. Enselme

Plan

- Descripteur de déploiement (web.xml)
- MVC – Modèle-Vue-Contrôleur

Descripteur de déploiement

- Le descripteur de déploiement d'une application web est un fichier nommé **web.xml** et situé dans le répertoire WEB-INF du répertoire racine de l'application web (WebContent).
- Il contient les caractéristiques et paramètres de l'application. Cela inclut la **description des servlets utilisées**, ou les différents **paramètres d'initialisation**.

Paramètres de contexte (1/2)

Dans web.xml (WebContent/WEB-INF)

```
<!-- Titre de l'application -->
```

```
<context-param>
```

```
    <param-name>title</param-name>
```

```
    <param-value>pierre-papier-ciseaux</param-value>
```

```
</context-param>
```

```
<!-- URLs communes aux vues -->
```

```
<context-param>
```

```
    <param-name>entetedepage</param-name>
```

```
    <param-value>/WEB-INF/JSP/commun/entetedepage.jsp</param-value>
```

```
</context-param>
```

```
...
```

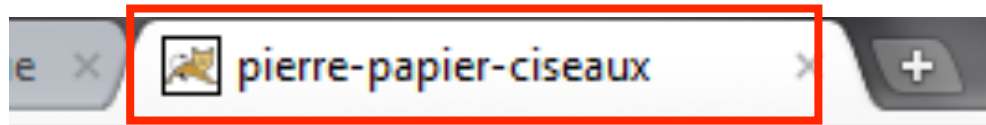
Paramètres de contexte (2/2)

– Accès aux paramètres dans une servlet

```
urlEntete = getServletContext().getInitParameter("entetedepage");
```

– Accès aux paramètres dans une JSP

```
<title><%= getServletContext().getInitParameter("title")%></title>
```



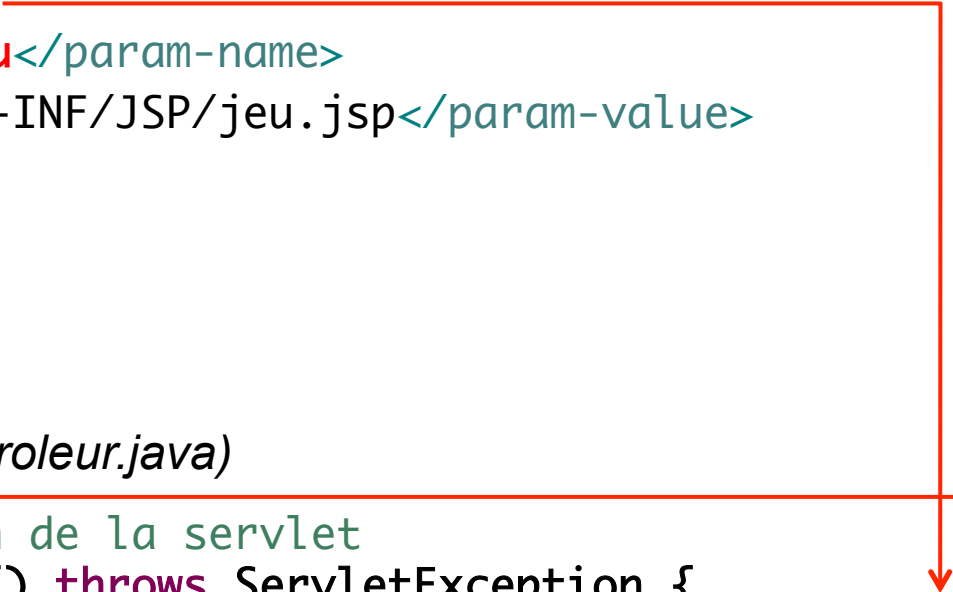
Description d'une servlet (1/2)

Dans web.xml (WebContent/WEB-INF)

```
<!-- Servlet controleur -->
<servlet>
  <servlet-name>controleur</servlet-name>
  <servlet-class>jeu.controleur.Controleur</servlet-class>
  <init-param>
    <param-name>urlJeu</param-name>
    <param-value>/WEB-INF/JSP/jeu.jsp</param-value>
  </init-param>
  ...
</servlet>
```

Dans la servlet (Controleur.java)

```
// Initialisation de la servlet
public void init() throws ServletException {
    urlJeu = getServletConfig().getInitParameter("urlJeu");
    ...
}
```



Description d'une servlet (2/2)

- Mapping URL / Servlet

Dans web.xml (WebContent/WEB-INF)

```
<servlet-mapping>  
    <servlet-name>contrôleur</servlet-name>  
    <url-pattern>/do/*</url-pattern>  
</servlet-mapping>
```

- Accès à la servlet depuis une URL

Dans un fichier HTML ou JSP

```
<a href="<%= getServletContext().getContextPath()%>/do/jeu">rejouer</a>  
<form method="post" action="<%= getServletContext().getContextPath()%>/do/  
resultat">
```

Plus besoin de l'annotation @WebServlet("...")

Description d'une JSP (1/3)

- Même fonctionnement !
- Rappel : **une page JSP EST une servlet**

Description d'une JSP (2/2)

Dans web.xml (WebContent/WEB-INF)

```
<servlet>  
  <servlet-name>JSPjeu</servlet-name>  
  <jsp-file>/jeu/jeu.jsp</jsp-file>  
  <init-param>  
    <param-name>JSPparametre</param-name>  
    <param-value>v2</param-value>  
  </init-param>  
</servlet>
```

Dans la JSP (jeu.jsp)

```
<h3>Choisissez une main (  
<%= getServletConfig().getInitParameter("JSPparametre") %>)  
</h3>
```

Pierre-Papier-Ciseaux

Choisissez une main (v2)

Descripteur de déploiement

- Autres propriétés
 - Description de l' application
 - <display-name><description>
 - Description des servlets
 - <servlet-class><description><load-on-startup>
 - Attributs de session
 - <session-config><session-timeout>
 - Fichiers index
 - <welcome-file-list><welcome-file>
 - Sécurité, pages d' erreurs, références d' EJB, etc.
 - Plus d' info: <http://www-igm.univ-mlv.fr/~dr/XPOSE2003/tomcat/tomcat.php?rub=16>

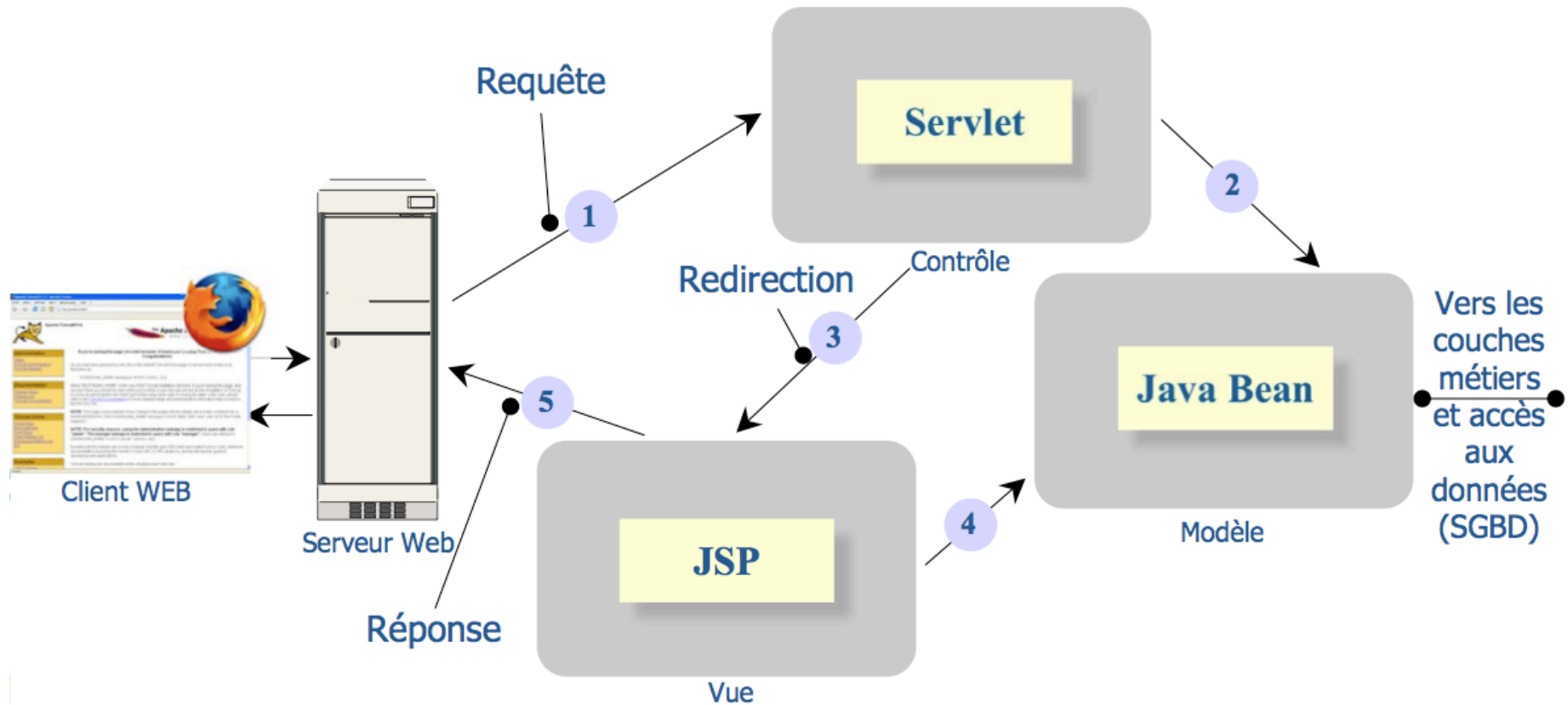
Conclusion des premières séances

- Les pages **JSP** doivent **contenir le moins de java possible**
 - Besoin de **structurer le code** plus finement **selon les compétences de chaque technologie** (JSP, Servlet, etc.)
- ⇒ Utilisation du modèle **MVC** : Modèle - Vue - Contrôleur

MVC (1/3)

- Architecture MVC (en règle générale)
 - Le **contrôle** est géré par les **Servlets**
 - Accède au modèle
 - Redirige vers la vue
 - Le **modèle** est géré par les **Beans**
 - La **vue** est géré par les **JSP**

MVC (2/3)



MVC (3/3)

- Besoin de savoir communiquer entre des Servlets et des JSP
 - Requête envoyée par le client (1)
 - Communication entre le contrôle et le modèle (2)
 - Transmission à la page JSP des informations du Java Bean (3)
 - La page JSP utilise les informations du JavaBean traduit (4)
 - La vue est envoyée en réponse au client (5)

JSP, Servlet et MVC

- Dans la Servlet

Instance du Bean
envoyée comme attribut

Ressource à rediriger

```
bean = new MonBean(...);  
req.setAttribute("idbean", bean);  
RequestDispatcher dispatch = this.getServletContext().getRequestDispatcher("/page.jsp");  
dispatcher.forward(req, res);
```

- Dans la JSP

Même nom que l'attribut
donné dans la Servlet

```
<jsp:useBean id="idbean" class="monpackage.MonBean" scope="request"/>  
<%= idbean... %>
```

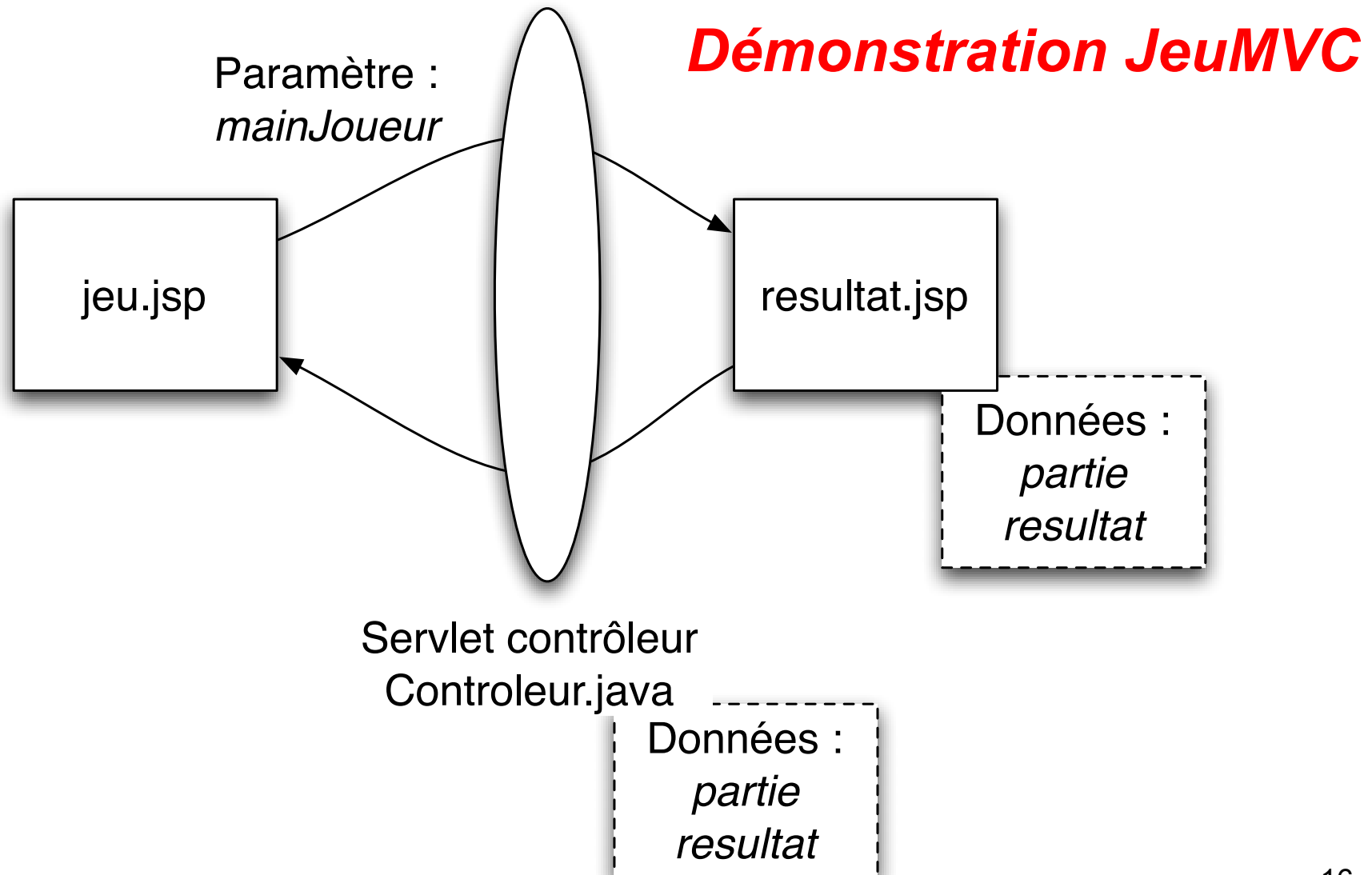
OU

```
<%= monpackage.MonBean idbean = (monpackage.MonBean) request.getAttribute("idbean"); %>  
<%= idbean... %>
```

```
<%@ page import="monpackage.MonBean" %>  
<% MonBean idbean = (MonBean) request.getAttribute("idbean"); %>
```

Pour éviter de mettre les
noms de package
utilisation de la directive
page import

Jeu version MVC



Un contrôleur pour les gérer tous

```
public class Controleur extends HttpServlet {  
    ...  
    public void doGet(HttpServletRequest request, HttpServletResponse response)  
        throws IOException, ServletException {  
        ...  
        // On récupère l'action à exécuter  
        String action = request.getPathInfo();  
        ...  
        // Exécution action  
        if (methode.equals("get") && action.equals("/jeu")) {  
            doJeu(request, response);  
  
        } else if (methode.equals("post") && action.equals("/resultat")) {  
            doResultat(request, response);  
  
        } else { // Autres cas  
            doJeu(request, response);  
        }  
    }  
}
```

Rappel : javabeau et session (1/2)

- Enregistrer les données d'un utilisateur pendant un temps donné

```
request.getSession().setAttribute("resultat", resultat);
```

- Une session par utilisateur (par instance de navigateur)
- Une session est commune à toutes les servlets

Rappel : javabeans et session (2/2)

- Dans Resultat.jsp

```
// Récupération des java beans
<jsp:useBean id="jeu" class="...Jeu" scope="request"/>
<jsp:useBean id="resultat" class="...Resultat" scope="session"/>
...
// Utilisation des java beans
<p>nombre de victoires : <jsp:getProperty name="resultat"
    property="nombreVictoire" /></p>
<p>nombre de défaites : <jsp:getProperty name="resultat"
    property="nombreDefaite" /></p>
<p>nombre d'égalités : <jsp:getProperty name="resultat"
    property="nombreEgalite" /></p>
```

MVC : sécurité

- Contrôleur vérifie les données en entrée
- JSP dans le dossier WEB-INF
 - Accès direct aux JSP interdit
- JSP en paramètre de la servlet contrôleur (web.xml)
 - Seules ces JSP seront utilisées

Pour aller plus loin

- Les bases du développement web MVC en Java par Serge Tahé

<http://tahe.developpez.com/java/baseswebmvc/>

Votre projet *projet_SIL4.pdf* sur Chamilo

- Etape 3 : **mise en place du MVC** (cours 3)
 - Création d'un contrôleur pour gérer l'application, intégration des pages réalisées dans la première étape, création des pages JSP d'édition : notes, absences
 - Rappel : vous pouvez utiliser le framework CSS bootstrap pour une mise en page HTML rapide (<http://getbootstrap.com>).
- Etape 4 : mise en place de la persistance (cours 4)