Formation Java EE

Web - JSP

digitalent FORMATION

JSP

Pourquoi JSP ?

- Une page HTML avec l'API Servlet ?

```
public class MaServletWeb extends HttpServlet {

  @Override
  protected void doGet(HttpServletRequest req,
        HttpServletResponse resp)
        throws ServletException, IOException {

    resp.setContentType("text/html");
    PrintWriter out = resp.getWriter();
    out.println("<!DOCTYPE html>");
    out.println("<html>");
    out.println(" <head>");
    out.println("   <title>Mon titre</title>");
    out.println(" </head>");
    out.println(" <body>");
    out.println("   <h1>Ma super page</h1>");
    out.println("   <p>La date du jour => "+new Date()+"</p>");
    out.println(" </body>");
    out.println("</html>");

  }

}
```

JSP

Pourquoi JSP ?

- Améliorer la productivité

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Mon titre</title>
</head>
<body>
  <h1>Ma super page</h1>
  <p>
    La date du jour => <%=new java.util.Date()%>
  </p>
</body>
</html>
```

```
public class MaServletWeb extends HttpServlet {

  @Override
  protected void doGet(HttpServletRequest req,
        HttpServletResponse resp)
        throws ServletException, IOException {

    resp.setContentType("text/html");
    PrintWriter out = resp.getWriter();
    out.println("<!DOCTYPE html>");
    out.println("<html>");
    out.println(" <head>");
    out.println("   <title>Mon titre</title>");
    out.println(" </head>");
    out.println(" <body>");
    out.println("   <h1>Ma super page</h1>");
    out.println("   <p>La date du jour => "+new Date()+"</p>");
    out.println(" </body>");
    out.println("</html>");

  }

}
```

JSP

Tags JSP

- Permettent de différencier le code HTML du code Java
- **<%@ … %>** // tags de directive
- **<%-- … --%>** // tags de commentaire
- **<%! … %>** // tags de déclaration
- **<% … %>** // tags de script
- **<%= … %>** // tags d'expression

JSP

Directives

- Les directives contrôlent comment le serveur Web doit générer la Servlet
- Elles sont placées entre les symboles **<%@** et **%>**
- Les directives suivantes sont disponibles
  - **include** : indique au compilateur d'inclure un autre fichier

```
<%@ include file="entete.jsp" %>
```

  - **taglib** : indique une bibliothèque de balises à utiliser

```
<%@ taglib prefix="monprefix" uri="taglib/montag.tld" %>
```

  - **page** : définit les attributs spécifiques à la page

```
<%@ page contentType="text/plain" %>
```

JSP

Commentaires

- Ils sont placées entre les symboles **<%--** et **--%>**
- Le texte en commentaire ne sera pas envoyé au client, ni compilé dans la servlet

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Mon titre</title>
</head>
<body>
  <%-- Ce texte ne s'affichera pas --%>
  Ce texte s'affichera

</body>
</html>
```

JSP

Déclarations

- Les déclarations sont placées entre **<%!** et **%>**
- Utilisés pour compléter la structure de classe (ajout d'attributs ou ajouts / redéfinition de méthodes)

```
<%!
   private int compteur = 0;

   private int incrementer() {
     return compteur++;
   }
%>
```

JSP

Scripts

- Les scripts sont placés entre les symboles **<%** et **%>**
- Les scripts sont composés de code Java qui ont accès aux attributs / méthodes définis par le tag déclaration (<%! … %>) et à des objets implicites

```
<%
   for (int i=0;i<10;i++) {
%>
  <h1>Ce texte s'affichera 10 fois</h1>
<%
   }
%>
```

JSP

Expression

- Les expressions sont placés entre les symboles **<%=** et **%>**
- Permettent d'évaluer une expression et renvoyer sa valeur

```
<%
   for(int i=0;i<10;i++) {
%>
  <h1>Ce texte s'affichera 10 fois. Itération i= <%= i %></h1>
<%
   }
%>
```

JSP

Objets implicites

- **request** : objet requête
- **response** : objet réponse
- **session** : session courante
- **out** : flux de sortie de la réponse
- **application** : contient des méthodes log() permettant d'écrire des messages dans le journal du contenu (ServletContext)
- **pageContext** : utilisé pour partager directement les variables entre pages JSP
- **exception** : disponible uniquement dans les pages erreurs donnant des informations sur les erreurs

JSP

Gestion des erreurs

- Une page JSP peut indiquer la page à afficher en cas d'erreurs
- Les erreurs sont déclenchées par l'intermédiaire des exceptions qui seront transmises à la page d'erreur
- Exemple de déclaration d'une page d'erreur

```
<%@ page errorPage="pageErreur.jsp" %>
```

JSP

Gestion des erreurs

- Une page JSP est définie comme une page d'erreur par la directive **page** et l'attribut **isErrorPage**

```
<%@ page isErrorPage="true" %>
```

- L'exception ayant causée l'erreur est accessible via l'objet **exception**.

JSP

Java Beans et JSP

- Pour déclarer et allouer un Bean Java dans une page JSP, il faut utiliser l'action **<jsp:useBean>**

```
<jsp:useBean id="maPizza" class="fr.pizzeria.Pizza" scope="request" />
```

- **id** : nom de l'instance
- **class**: le nom complet de la classe
- **scope**: portée de l'objet créé. Peut prendre les valeurs :
  - request
  - page
  - session
  - application

JSP

Java Beans et JSP

- Pour lire la valeur de la propriété d'un bean :

```
<jsp:getProperty name="maPizza" property="nom" />
```

- Pour mettre à jour la valeur de la propriété d'un bean :

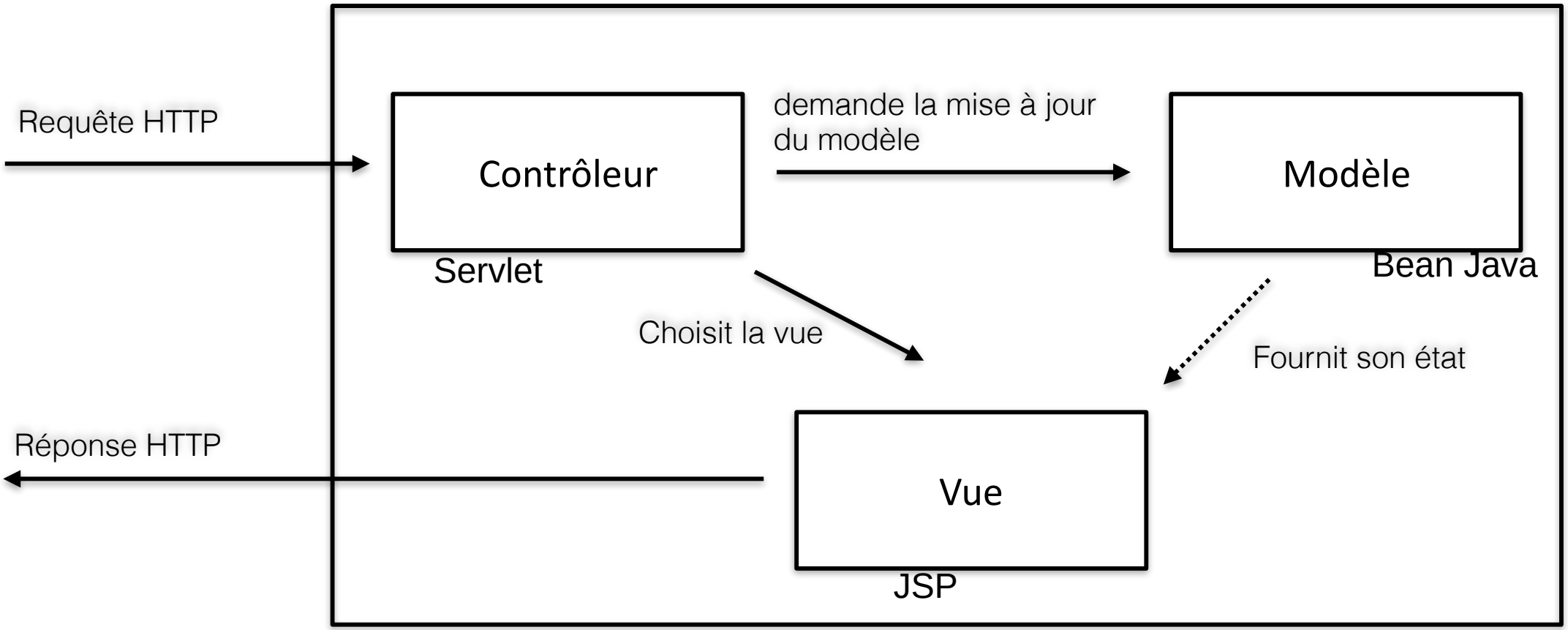
```
<jsp:setProperty name="maPizza" property="nom" property="bob" />
```

MVC

MVC ?

- Organiser, structurer une application interactive en séparant :
  - Les données et leurs traitements : **Modèle**
  - La représentation des données : **Vue**
  - Le comportement de l'application : **Contrôleur**

MVC

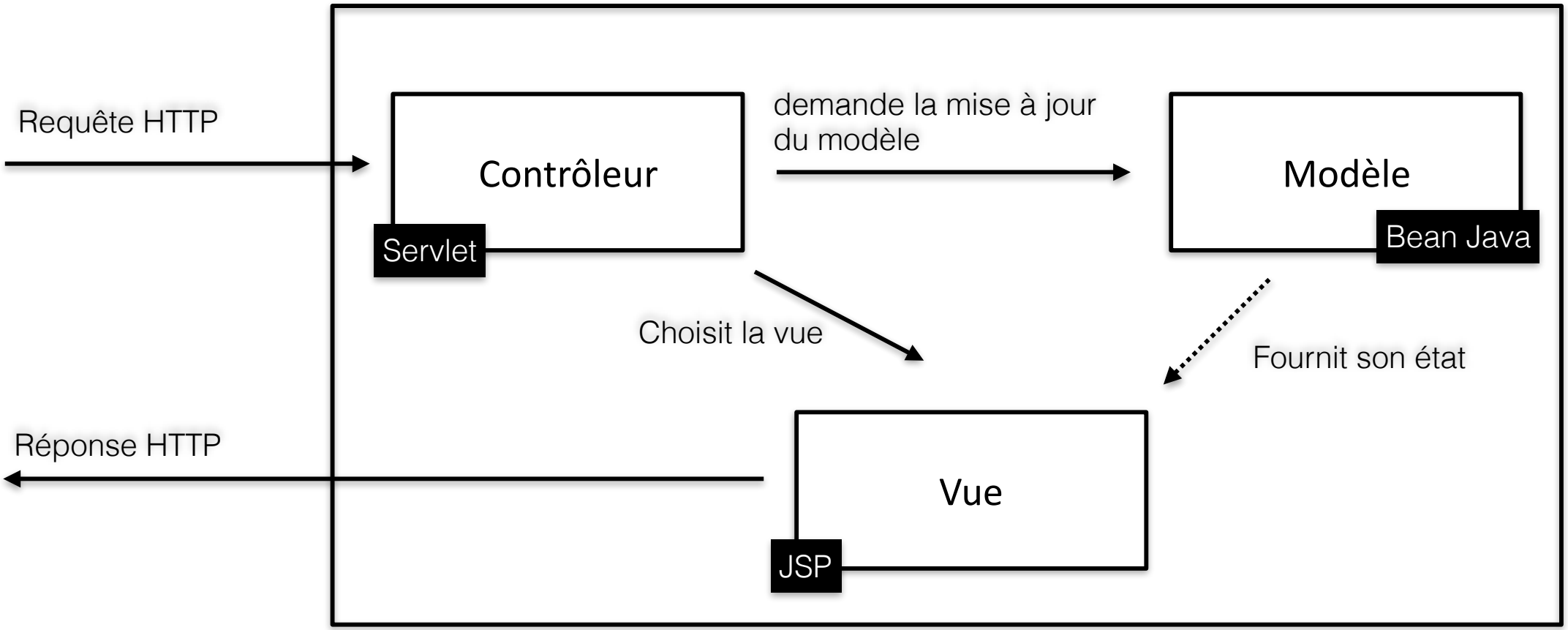


MVC

MVC avec Servlet et JSP

- Il est possible d'appliquer les principes MVC en suivant l'organisation suivante :
  - Contrôleur —> Servlet
  - Vue —> JSP
  - Modèle —> Bean Java

MVC



MVC

Rediriger une requête vers une vue

- La classe RequestDispatcher permet de faire évoluer la direction d'une requête au sein du serveur.
- A ne pas confondre avec une redirection HTTP, le client HTTP n'est pas au courant du changement de routage.

```
RequestDispatcher dispatcher = this.getServletContext()
                    .getRequestDispatcher("/unePage.jsp");

dispatcher.forward(req, resp);
```