

TD n°3 Java EE

- [Sujet TD n°3](#)
- [Archive Web à importer dans Eclipse](#)
- [Librairie Jar du projet en mode console](#)
- [Zip du projet en mode console](#)
- [Documentation des classes Web](#)
- [Documentation des classes du mode console](#)

Classes

Servlet Controller

- Créer la servlet **MainController** dans le package **web.controller**

```
/**
 * Servlet implementation class MainController
 */
@WebServlet("*.do")
public class MainController extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * Retourne le flux de sortie (de quoi écrire sur la page)
     * @param response réponse HTTP
     * @return le flux de sortie out
     * @throws IOException
     */
    private PrintWriter getOut(HttpServletResponse response) throws IOException{
        response.setCharacterEncoding("UTF-8");
        PrintWriter out=response.getWriter();
        return out;
    }
    /**
     * Test la méthode de la requête
     * @param request
     * @return vrai si la méthode est POST
     */
    private boolean isPost(HttpServletRequest request){
        return request.getMethod().equalsIgnoreCase("post");
    }
    /**
     * Retourne la session active
     * @param request requête HTTP
     * @return la session en cours
     */
    private HttpSession getSession(HttpServletRequest request){
        return request.getSession();
    }
    /**
     * Retourne l'action sollicitée par la requête
```

```

    * @param request Requête Http
    * @return action demandée
    */
    private String getAction(HttpServletRequest request){
        String result="";
        String[] parts=request.getRequestURI().split("/");
        if(parts.length>0)
            result=parts[parts.length-1];
        if(request.getContextPath().contains(result))
            result="index";
        result=result.replace(".do","");
        return result;
    }
    /**
    * Charge une vue (jsp située dans le dossier protégé WEB-INF)
    * @param viewName nom de la vue
    * @param request Requête Http
    * @param response Réponse HTTP
    * @throws ServletException
    * @throws IOException
    */
    private void loadView(String viewName,HttpServletRequest
request,HttpServletResponse response){
        try {
            request.getRequestDispatcher("WEB-INF/"+viewName).include(request,
response);
        } catch (ServletException | IOException e) {
            e.printStackTrace();
        }
        System.out.println("redirection vers WEB-INF/"+viewName);
    }
    /**
    * Exécute l'action actionName en appelant la méthode du même nom de la servlet
    * @param actionName nom de l'action à exécuter
    * @param request requête HTTP
    * @param response réponse HTTP
    * @return Vrai si l'action a été trouvée
    */
    private boolean executeAction(String actionName,HttpServletRequest request,
HttpServletResponse response){
        Method method=null;
        boolean result=true;
        try {
            method=this.getClass().getMethod(actionName,
HttpServletRequest.class,HttpServletResponse.class);
        } catch (NoSuchMethodException | SecurityException e) {
            System.out.println("Action "+actionName+" inexistante");
            result=false;
        }
        if(method!=null){
            try {
                method.invoke(this, request,response);
            } catch (IllegalAccessException | IllegalArgumentException |
InvocationTargetException e) {
                System.out.println("Erreur d'exécution de l'action "+actionName +"
: "+e.getMessage());
            }
        }
    }

```

```

        }
    }
    return result;
}
/**
 * @see HttpServlet#HttpServlet()
 */
public MainController() {
    super();
    // TODO Auto-generated constructor stub
}

/**
 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse
response)
 */
protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
    String action=getAction(request);
    if(action!=null && !"".equals(action)){
        if(!executeAction(action, request, response)){
            //TODO Implémenter si action inexistante
        }
    }
}

/**
 * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse
response)
 */
protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
    doGet(request, response);
}
public void index(HttpServletRequest request,HttpServletResponse response){
    loadView("index.jsp", request, response);
}
//TODO Autres actions à implémenter
}

```

Exemple d'implémentation

Gestion de la suppression d'un utilisateur

Vue frmDeleteUser

```

<form name="frmDelete" id="frmDelete">
    <%
        SessionApp sessionApp=Utils.getSessionApp(session);
        out.print(ConvertToHtml.mapToSelect(sessionApp.getApplication().getLesUtilisateurs(
), "txtLogin",10));
    %>

```

```
</form>
<input type="button" id="btDeleteUser" class="btn" value="Supprimer">
```

Méthode du contrôleur deleteUser

```
/**
 * Gestion de la suppression d'un utilisateur
 * @param request
 * @param response
 * @throws IOException
 */
public void deleteUser(HttpServletRequest request, HttpServletResponse
response) throws IOException{
    PrintWriter out=getOut(response);
    if(isPost(request)){
        SessionApp sessionApp=Utils.getSessionApp(request);
        sessionApp.getApplication().setView(new HtmlView(out));
        Commande cmd=new CmdDeleteUser(sessionApp);
        cmd.addArg(request.getParameter("txtLogin"));
        cmd.run();
        out.print(JSFormsUtils.get("/deleteUser.do", "", "divFrmCommande"));
    }else{
        loadView("forms/frmDeleteUser.jsp", request, response);
        out.print(JSFormsUtils.postFormAndBindTo("#btDeleteUser", "click",
"/deleteUser.do", "frmDelete", "divResultCommande"));
    }
}
```

Éléments de correction

Script.js

Associer les items du menu aux actions en *.do en modifiant le fichier script.js :

```
run=function(obj){
    cmd=obj.id;
    new Forms.Ajax('divFrmCommande',cmd+'.do').get();
    $("divResultCommande").innerHTML=obj.title;
};
```

main.jsp

Modifier les ids des éléments du menu, pour les faire correspondre aux actions (méthodes du contrôleur) :

```
<div id="grp-utilisateurs" class="categorie">Utilisateurs</div>
<ul id="grp-child-utilisateurs">
    <li id="addUser" class="item" title="Entrer le login de
l'utilisateur à créer">Ajouter</li>
```

```
<li id="frmUpdateUser" class="item" title="Sélectionner  
l'utilisateur à modifier">Modifier</li>  
<li id="frmDeleteUser" title="Sélectionner l'utilisateur à  
supprimer" class="item">Supprimer</li>  
<li class="item">Affecter à ...</li>  
<li class="item">Liste</li>
```

L'item addUser pour ajouter un utilisateur correspond à la méthode du contrôleur :

```
public void addUser(HttpServletRequest request, HttpServletResponse response){  
}
```

Ajout d'un utilisateur

```
/**  
 * Ajout d'un utilisateur  
 * @param request  
 * @param response  
 * @throws IOException  
 */  
public void addUser(HttpServletRequest request, HttpServletResponse response)  
throws IOException{  
    PrintWriter out=getOut(response);  
    if(isPost(request)){  
        SessionApp  
sessionApp=(SessionApp)request.getSession().getAttribute("sessionApp");  
        sessionApp.getApplication().setView(new HtmlView(out));  
        Commande cmd=new CmdAddUser(sessionApp);  
        cmd.addArg(request.getParameter("txtLogin"));  
        cmd.run();  
    }else{  
        loadView("forms/frmAddUser", request, response);  
        out.print(JSFormsUtils.postFormAndBindTo("#btAddUser", "click",  
"actions/doAddUser.jsp", "frm", "divResultCommande"));  
    }  
}
```

From:

<http://slamwiki2.kobject.net/> - **SlamWiki 2.1**

Permanent link:

<http://slamwiki2.kobject.net/javaee/td3?rev=1384852254>

Last update: **2019/08/31 14:42**

