**ENST Bretagne**
Département Informatique

**TP Servlet**
**Utilisation des cookies, d'une session, accès à une base de données et utilisation de filtres**

# Gestion d'une session lors d'une commande dans un magasin

## 1 Objectifs :

Une application sur le WEB se présente comme une succession de pages, en principe il n'y a aucun lien entre toutes ces pages, le serveur ne connaît le client que le temps d'une requête, or dans diverses applications (comptabilité commerce électronique, gestion de comptes bancaires, etc.) il est nécessaire de factoriser un certain nombre de données propres à chaque client entre ces différentes pages.

Ce TP, par l'utilisation des cookies et des sessions http et d'accès à une base de données va montrer les différentes possibilités.

## 2 Installation d'éclipse

- **Cette installation est déjà faite sur les machines virtuelles**

**Eclipse** est un environnement de programmation, orienté Java, qui peut s'adapter par des plugins à différents contextes et langages, ici nous allons l'adapter à la langue française et à tomcat.

1) Extraire les fichiers de **eclipse-jee-europa-fall2-win32.zip** dans **c :/jsp**.

Créez un répertoire "**travail**" sous C:, c'est ce répertoire que vous donnerez au lancement d'éclipse comme "**Workspace**". Lancer eclipse par la commane **eclipse.exe** sous le répertoire eclipse. qui a été créé.

Vous pouvez observer qu'il ne connaît pas tomcat, pas de menu "**tomcat**". Fermer le.

- **Le plugin tomcat**

4) Extraire les fichiers de **tomcatPluginV321.zip**dans **c :/jsp/eclipse/plugins**

Relancez **eclipse,** vous devez voir un menu **tomcat** qui permet de lancer le serveur et de l'arrêter.

Quelques initiations sous eclipse

1) sous le menu "**Window**" allez sous "**preference**". Dans "**preference**", ouvrez "**java**" et sélectionnez "**installed JREs**". Vérifiez que la JRE choisie est celle que vous avez installée, sinon rajoutez votre JRE.

2) sous le menu "**Window**" allez sous "**preference**". Dans "**preference**", ouvrez "**tomcat**" et sous **tomcat** mettre la version "6.x", mettre (avec browse) le répertoire où se trouve votre tomcat,

en principe "**C:\jsp\tomcat6.0**". Vérifiez que pour les contextes, vous avez bien un fichier par contexte et que ce fichier se trouve : "**C:\jsp \Tomcat 6.0\conf\Catalina\localhost**". Ceci veut dire qu'Eclipse mettra dans ce répertoire un fichier de contexte pour chaque application WEB que vous créerez. Par la suite, vous pourrez regarder ce fichier.

3) sous le menu "**Window**" allez sous "**preference**". Dans "**preference**", ouvrez "**tomcat**" et choisissez **"Paramètrage de la JVM",** vérifier que la JRE est la bonne, dans le dernier cadre "Boot Classpath" rajouter **rt.jar** que vous trouvez sous le répertoire **lib** de la machine virtuelle java.

## 3 Test d'une application

Nous allons voir comment créer une application web et la servlet suivante avec éclipse. Ce fichier se trouve dans les données communes.

1) sous le menu "**File**" allez sous "**new**" et choisissez "**Project**" . Dans " **Project** ", ouvrez "**java**" et choisissez **"Projet Tomcat"** puis **"next".** Donnez un nom à votre projet par exemple "monprojet" et "**Finish**".

Eclipse crée une application web, vous pouvez regarder le contexte qu'il a mis dans le fichier "monprojet.xml" sous **C:\jsp \Tomcat 6.0\conf\Catalina\localhost** vous voyez qu'il y a un attribut **reloadable="true"**, qui veut dire que tomcat doit recharger votre servlet si elle est modifiée.

Lancer tomcat par l'icône tomcat qui apparaît sous eclipse, sous un navigateur web, regardez si votre application apparaît dans le manager de tomcat.

2) sélectionnez votre projet sous eclipse et faite "**new**" et choisissez "**class**", donnez lui un nom (Majuscule en premier caractère), par exemple "Maservlet" et la superclass **javax.servlet.http.HttpServlet** et "**Finish**".

3) Ouvrir cette classe et faire un copier- coller avec la classe suivante, bien sûr changer lui son nom par "Maservlet" si vous avez gardé ce nom..

```
public class SimpleServlet extends HttpServlet  {
      public void doGet (HttpServletRequest request,
                         HttpServletResponse response)    throws ServletException, IOException  {
          PrintWriter       out;
          String            title = "Simple Servlet Output";
          response.setContentType("text/html");
          out = response.getWriter();
          out.println("<HTML><HEAD><TITLE>");
          out.println(title);
          out.println("</TITLE></HEAD><BODY>");
          out.println("<H1>" + title + "</H1>");
          out.println("<P>This is output from SimpleServlet.");
          out.println("</BODY></HTML>");
          out.close();
      }  }
```

4) Acceptez les "**import**" qu'éclipse vous propose pour corriger les erreurs.

5) Enregistrez votre servlet

6) Si dans le fichier "web.xml" du répertoire "tomcat6.0/conf" vous avez retiré les commentaires autour de "Invoker", vous pouver lancer cette servlet dans un navigateur par l'adresse **http://localhost:8080/monpremier/servlet/Maservlet.**

7) modifiez la servlet , enregistrez la et relancez la. La modification doit se voir (reloadable="true").

## 4 Travail demandé sur les servlets

- Le fichier web.xml, les fichiers Java se trouvent dans les données communes

### 4.1 Partie 1 : servlet : InscriptionClient .class alias : **servlet/sinscrire**

Un client pourra passer une commande que lorsqu'il possédera un cookie avec son nom et un cookie avec son mot de passe.

**Cas 1 :** Au premier appel de la servlet **InscriptionClient ,** il ne possède pas de cookies, et il n'y a pas de paramètres nom et motdepasse. La servlet lui envoie donc un formulaire pour lui demander de s'inscrire avec son nom et un mot de passe, ce formulaire rappelle la servlet avec les paramètres "nom" et "motdepasse".

**Cas 2 :** Au deuxième appel de la servlet **InscriptionClient** il ne possède pas toujours de cookies mais les paramètres nom et motdepasse sont présents , les cookies correspondant sont donc créés et un autre rappel est fait à cette même servlet et ceci sans paramètre (un seul cookie peut suffire, son nom le nom passé en paramètre, sa valeur le mot de passe).

**Cas 3 :** Au troisième appel, les cookies sont présents, mais il n'y a pas les paramètres (nom et mot de passe), la servlet lui envoie donc un formulaire pour qu'il s'identifie, ce formulaire rappelle la servlet avec le nom et le mot de passe.

**Cas 4 :** Au quatrième appel de la servlet, si les informations des cookies sont identiques à celles des paramètres (nom et motdepasse), le client peut accéder à la page de la servlet "**achat**".

```
package mesCommandes;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class InscriptionClient extends HttpServlet {
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
            String nomRecu=null, motPasseRecu=null;
            String nomCookie=null, motPasseCookie=null;

            // initialisation cookies et paramètres

            response.setContentType("text/html");
            PrintWriter out = response.getWriter();


   if (nomCookie==null && nomRecu==null){
                    // Cas 1 : cas où il n'y a ni de cookies ni de parametres
                out.println("<html>");
                out.println("<body>");
                out.println("<head>");
                out.println("<title>  inscription d'un client  </title>");
                out.println("</head>");
                out.println("<body bgcolor='white' >");
                out.println( nomRecu +" | "+ motPasseRecu +" | "+ nomCookie +" | "+ motPasseCookie );
                out.println("<h3>" + "Bonjour,  vous devez vous inscrire " + "</h3>");
                out.println("<h3>" + "Attention mettre  nom et le mot de passe avec plus de 3 caracteres" + "</h3>");
```

```
            out.print(" <form action='sinscrire' method='GET' > ");
            out.println("nom");
            out.println("<input type='text' size='20' name='nom'  >");
            out.println("<br>");
            out.println("mot de passe");
            out.println("<input type='password' size='20' name='motdepasse'> <br>");
            out.println("<input type='submit' value='inscription'>");
            out.println("</form>");
            out.println("</body>");
            out.println("</html>");
        } else  if (nomCookie==null && nomRecu!=null){

          // Cas 2 : cas  où il n'y a pas de cookies mais  les paramètres nom et mot de passes sont présents :

      }
          else if (identique(nomRecu,nomCookie) && identique(motPasseRecu,motPasseCookie))
                  {
                    // Cas 4 : cas où le nom et le mot passe sont correctes, appel à la servlet achat

  }
              else  {
                  // Cas 3 : les cookies sont présents demande de s'identifier
          }
    }

     public void doPost(HttpServletRequest request,
                  HttpServletResponse response)
     throws IOException, ServletException
    {
      doGet(request, response);
    }

     boolean identique (String recu, String cookie) {
      return ((recu != null) && (recu.length() >3) && (cookie != null) && (recu.equals(cookie) ));
    }
}
```

### 4.2 Partie 2 : Gestion d'une commande

Les servlets de gestion de la commande de disques sont les suivantes :

- **AfficherLesDisques.class** alias **servlet/achat**
- **CommanderUnDisque .class** alias **servlet/commande**

Au départ, l'accès à ces classes sera libre, par la suite, nous les ferons précéder par un filtre qui renverra à l'inscription si les cookies ne sont pas présents.

- **AfficherLesDisques.class**, cette servlet propose, une liste de disques où le client peut choisir celui qu'il va acheter, elle fournit pour chaque disque son titre et son prix, ainsi que la proposition de l'ajouter dans son panier d'achat (session).
  - La liste des disques et leur prix, pourraient être recherchés dans une base de données, pour l'instant, nous allons tout simplement utiliser une classe **Stock.** Cette classe contient une méthode **affiche** qui met dans un tableau HTML la liste des disques, leur prix, et un lien sur la servlet **CommanderUnDisque** avec comme paramètre le code du disque choisi. Au lieu de rechercher cette liste dans une base de données, elle est prise dans un tableau de disques créé dans cette classe.
  - Par la suite il sera possible de remplacer ce tableau par un accès à une base de données mais ce travail ne vous est pas demandé.

```
package mesCommandes;
```

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class AfficherLesDisques extends HttpServlet {

 public void doGet(HttpServletRequest request, HttpServletResponse response)  throws IOException, ServletException {
            Stock uneVente = new Stock();
            String nom = null;
            Cookie[] cookies = request.getCookies();
              nom = Identification.chercheNom(cookies);
              response.setContentType("text/html");
              PrintWriter out = response.getWriter();
              out.println("<html>");
              out.println("<body>");
              out.println("<head>");
              out.println("<title> Commande de disques </title>");
              out.println("</head>");
              out.println("<body bgcolor=\"white\">");
              out.println("<h3>" + "Bonjour " + nom + " vous pouvez commander un disque" + "</h3>");
              uneVente.vente(out);
              out.println("</body>");
              out.println("</html>");
 }

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

```
package mesCommandes;
import javax.servlet.http.*;

public class Identification {
      static String chercheNom (Cookie [] cookies) {
        // cherche dans les cookies la valeur de celui qui se nomme "nom"
        // retourne la valeur de ce nom au lieu de inconnu
    return "inconnu";
}
```

```
package mesCommandes;
import java.io.*;

public class Stock {
    static String [] [] leStock = {
            {"Disque CD - AMOR TICINES", "15", "disque897TR566"},
            {"Disque CD - Los Mayas", "19", "disque78UUNYT67"},
            {"Disque CD - Dick Anglas", "25", "disque87YHG564"},
            {"Disque CD - Frederic Angonas", "35", "disque98HUYU56"}
     };
     public static void vente (PrintWriter out) {
       out.println("<table border=1>");
       for(int i = 0; i< leStock.length; i++) {;
          out.println( "<tr> <td>" + leStock[i][0] + "    " + leStock[i][1] + " Euros  </td>");
          out.println(" <td><A HREF=\"commande?element=disque&code=" );
          out.println( leStock[i][2]+ "&ordre=ajouter&prix="   +leStock[i][1] + "\">");
          out.println("<IMG SRC=\"/fcexemple/images/panier.gif\" BORDER=0></A><br> </td> </tr>");
       }
       out.println("</table> </form>");
     }
}
```

- **CommanderUnDisque.class**, cette servlet, rajoute le disque dont le code est passé en paramètre dans le panier, c'est à dire dans la session, puis affiche les différents disques en cours de commande (leur code). Puis elle propose de commander un autre disque par un rappel à la servlet **AfficherLesDisques** ou d'enregistrer le contenu du panier dans une base de données par un appel à la servlet **EnregistrerCommande.**

```
package mesCommandes;
import java.io.*;
import java.text.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class CommanderUnDisque extends HttpServlet {
    public void doGet(HttpServletRequest request,
          HttpServletResponse response)
throws IOException, ServletException
{  String nom = null;
int nbreProduit = 0;
Cookie[] cookies = request.getCookies();
nom = Identification.verifier(cookies);

 response.setContentType("text/html");
 PrintWriter out = response.getWriter();
 out.println("<html>");
 out.println("<body>");
 out.println("<head>");
 out.println("<title>  votre commande </title>");
 out.println("</head>");
 out.println("<body bgcolor=\"white\">");
 out.println("<h3>" + "Bonjour  "+ nom + "  voici  votre commande" + "</h3>");
```

```
// affichage de tous les disques  présents dans le panier (éléments de la session)
// si parametre   ordre ==  ajouter     affichage du disque à ajouter au panier
//                                      ajout du nouveau disque dans le panier
```

```
out.println("<A HREF=achat> Vous pouvez commandez un autre disque </A><br> ");
 out.println("<A HREF=enregistre> Vous pouvez enregistrer votre commande </A><br> ");
 out.println("</body>");
 out.println("</html>");
}

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

### 4.3 Partie 2 : Enregistrement du panier dans une base de données

- **EnregistrerCommande.class** alias **servlet/enregistre**
- **ViderPanier.class** alias **servlet/vider**

- **EnregistrerCommande.class**, cette servlet, enregistre le contenu du panier dans une base de données. Si le client est inconnu dans la base, il est d'abord rajouté à la base. Puis, le client peut choisir entre terminer ses achats ou continuer. S'il veut continuer, on vide le panier dans la servlet **VidePanier** qui appelle la servlet "**achat**".

```
package mesCommandes;
  import java.io.*;
```

```
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;

public class EnregistrerCommande extends HttpServlet {
  Connection connexion=null;
  Statement stmt=null;
  PreparedStatement pstmt=null;

 public void doGet(HttpServletRequest request,   HttpServletResponse response)   throws IOException, ServletException
  {  String nom = null;
     int nbreProduit = 0;
     Cookie[] cookies = request.getCookies();
     boolean connu = false;
     nom = Identification. chercheNom  (cookies);
        OuvreBase();
        AjouteNomBase(nom);

      response.setContentType("text/html");
      PrintWriter out = response.getWriter();
      out.println("<html>");
      out.println("<body>");
      out.println("<head>");
      out.println("<title>  votre commande </title>");
      out.println("</head>");
      out.println("<body bgcolor=\"white\">");

      out.println("<h3>" + "Bonjour " + nom  + " voici ta nouvelle commande" + "</h3>");
      HttpSession session = request.getSession();
      Enumeration names = session.getAttributeNames();
      while (names.hasMoreElements()) {
         nbreProduit++;
         String name = (String) names.nextElement();
         String value = session.getAttribute(name).toString();
         out.println(name + " = " + value + "<br>");
       }
      AjouteCommandeBase(nom,session);
      out.println("<h3>" + "et voici " + nom  + "  ta commande  complete" + "</h3>");
      MontreCommandeBase(nom, out);

      out.println("<A HREF=mesCommandes.VidePanier> Vous pouvez commandez un autre disque </A><br> ");
      out.println("</body>");
      out.println("</html>");
   }

   protected void OuvreBase() {
    try {
      Class.forName("org.gjt.mm.mysql.Driver").newInstance();
      connexion = DriverManager.getConnection("jdbc:mysql://localhost/magasin","root","");
      connexion.setAutoCommit(true);
      stmt = connexion.createStatement();
    }
      catch (Exception E) {
           log(" -------- probeme  " + E.getClass().getName() );
           E.printStackTrace();
      }
  }

   protected void fermeBase() {
    try {
         stmt.close();
         connexion.close();
    }
     catch (Exception E) {
        log(" -------- probeme  " + E.getClass().getName() );
        E.printStackTrace();
      }
  }
```

```
        protected void AjouteNomBase(String nom) {
          try {
            ResultSet rset = null;
            pstmt= connexion.prepareStatement("select numero from personnel where nom=?");
            pstmt.setString(1,nom);
            rset=pstmt.executeQuery();
            if (!rset.next())
              stmt.executeUpdate("INSERT INTO personnel VALUES  (" ,'" +  nom + "' )" );
          }
          catch (Exception E) {
               log(" - probeme  " + E.getClass().getName() );
               E.printStackTrace();
           }
        }

        protected void AjouteCommandeBase(String nom, HttpSession session ) {

          // ajoute le contenu du panier   dans la base

        }

        protected void MontreCommandeBase(String nom, PrintWriter out) {

          // affiche les produits présents dans la base

        }

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

### 4.4 Partie 2 : Filtres de vérification d'accès aux pages

**1)** Lorsque ces servlets fonctionnent, vous faites précéder leur exécution par un filtre qui vérifie si le client est bien inscrit (cookies présents), ce filtre renvoie sur l'inscription en cas d'absence de cookies :

- **Filtre Autorisation.class** nom : **portier**

```
package mesCommandes;
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class MonFiltre implements Filter {
    private FilterConfig filterConfig = null;

    public void init(FilterConfig filterConfig) throws ServletException {
                this.filterConfig = filterConfig;
      }

 public void doFilter(ServletRequest request, ServletResponse response,  FilterChain chain) throws IOException,
 ServletException  {
     String nom = null;
     HttpServletRequest hrequest = (HttpServletRequest) request;
     HttpServletResponse hresponse = (HttpServletResponse) response;
        Cookie[] cookies = hrequest.getCookies();

          // test s'il existe un cookie dont l'attribut est "nom"
          If        {    // cas ou il n'existe pas appel de la servlet    inscrire
          }

          else {
          chain.doFilter(request, response); }
        }
```

```
        public void destroy() {
            this.filterConfig = null;
        }
    }
```

**2)** Ecrire le filtre **FiltrerechercheNom.class** qui permet aux sevlets de connaître le nom du clients sans avoir à chercher elle même dans le cookie.

- **Filtrerecherc heNom.class** nom : **initiation**

## 4.5 Remarque

Ce TP utilise une base de données, ce sera une base **Mysql,** le code serait le même pour une autre base à part le choix du driver.

La base s'appelle "magasin", elle contient deux tables : personnel et commande

La table "personnel" possède deux attributs une clé qui est un entier et le nom qui est une chaîne de caractère.

La table "commande" possède trois attributs une clé qui est un entier et le produit qui est une chaîne de caractère et une référence sur une clé de la table "personnel".

Voici la construction des tables :

```
# Base de données: `magasin`
# Structure de la table `commande`
#
CREATE TABLE commande (
  num tinyint(4) NOT NULL auto_increment,
  article varchar(20) NOT NULL default '',
  qui tinyint(4) NOT NULL default '0',
  PRIMARY KEY  (num)
) TYPE=MyISAM;
# --------------------------------------------------------

#
# Structure de la table `personnel`
#
CREATE TABLE personnel (
  numero tinyint(4) NOT NULL auto_increment,
  nom varchar(20) NOT NULL default '',
  PRIMARY KEY  (numero)
) TYPE=MyISAM;
```

Vous trouverez ces fichiers **sql** dans un répertoire **bd** des données.

### *4.5.1 Le CLASSPATH.*

La variable CLASSPATH doit contenir l'archive contenant les API sur les servlets si vous voulez compiler en dehors d'éclipse.

**C:/jsp/ *Tomcat 6.0* /lib/servlet-api.jar**

## 4.6 Accès à la base de données

Cet accès est possible si la librairie gérant le driver est connue de l'application WEB. Sous le répertoire WEB-INF de votre application vous rajoutez donc un répertoire **lib** qui contient une archive avec ce driver : **mysql-connector-java-3.0.17-ga-bin.jar,** pour accéder à une base de donnée mysql.

## 4.7 Accès à l'information sur les servlets et JSP

**Servlets**

http://java.sun.com/j2ee/tutorial/1_3-fcs/doc/Servlets.html

**JSP**

http://www.jspin.com/

http://www.apl.jhu.edu/~hall/java/Servlet-Tutorial/Servlet-Tutorial-JSP.html

http://java.sun.com/j2ee/tutorial/1_3-fcs/doc/JSPIntro.html#62069

http://tecfa.unige.ch/guides/tie/html/java-jsp/java-jsp.html

http://tecfa.unige.ch/guides/jsp/pointers.html

http://www.multimania.com/deneau/jsp.html

Et n'oublier pas le moteur de recherche **http://www.google.fr**

**HTML**

http://perso.wanadoo.fr/chatinais/index.htm

**JDBC**

http://java.sun.com/j2se/1.4.2/docs/guide/jdbc/getstart/GettingStartedTOC.fm.html

## 5 Compte rendu

Vous devez rendre l'ensemble des classes (l'application web entière) qui constituent ce TP, dans une archive

**nom1ETnom2Servlet.zip**

nom1 et nom2 sont les deux noms d'un binôme.

Vous pouvez commenter votre code !

Cette archive vous l'envoyez par mail en pièce jointe à **robert.ogor@enst-bretagne.fr**

Date limite **Lundi 25 février 2008**

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE web-app
    PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
    "http://java.sun.com/dtd/web-app_2_3.dtd">
<web-app>
    <display-name>exemple formation</display-name>
    <description>
       exemple formation servlet et jsp
    </description>
<!--
<filter>
     <filter-name>portier</filter-name>
     <filter-class>mesCommandes.MonFiltre</filter-class>
</filter>
-->
```

```
<!--
<filter-mapping>
      <filter-name>portier</filter-name>
         <servlet-name>acheter</servlet-name>
</filter-mapping>
<filter-mapping>
      <filter-name>portier</filter-name>
         <servlet-name>commander</servlet-name>
</filter-mapping>
-->
    <servlet>
        <servlet-name>inscrire</servlet-name>
        <servlet-class>mesCommandes.InscriptionClient</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>inscrire</servlet-name>
        <url-pattern>/servlet/sinscrire</url-pattern>
    </servlet-mapping>

    <servlet>
        <servlet-name>acheter</servlet-name>
        <servlet-class>mesCommandes.AfficherLesDisques</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>acheter</servlet-name>
        <url-pattern>/servlet/achat</url-pattern>
    </servlet-mapping>

     <servlet>
        <servlet-name>commander</servlet-name>
        <servlet-class>mesCommandes.CommanderUnDisque</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>commander</servlet-name>
        <url-pattern>/servlet/commande</url-pattern>
    </servlet-mapping>

     <servlet>
        <servlet-name>enregistrer</servlet-name>
        <servlet-class>mesCommandes.EnregistrerCommande</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>enregistrer</servlet-name>
        <url-pattern>/servlet/enregistre</url-pattern>
    </servlet-mapping>

   <servlet>
        <servlet-name>vider</servlet-name>
        <servlet-class>mesCommandes.VidePanier</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>vider</servlet-name>
        <url-pattern>/servlet/vider</url-pattern>
    </servlet-mapping>
</web-app>
```