133 - Réaliser des applications web en session-Handling

Rapport personnel

Date de création : 23.03.2023
Version 1 du jj.mm.aaaa

Luca Gambera

E — M — F
Ecole des Métiers
Berufsfachschule
Fribourg
Informatique

Module du 23.03.2023
au jj.mm.aaaa

Table des matières

1 Exercice test techno

1.1 Exercice TT 1

Durant l'exercice nous avons préparé le serveur Tomcat qui nous servira pour les prochains exercices. J'ai fait cela en suivant les étapes dans OneNote. J'ai aussi ajouté le serveur Tomcat dans NetBeans.

Ci-dessous vous pouvez voir des commandes utiles qui nous permet de voir les applications qui tourne sous le port 8080 et de pouvoir arrêter cette application.

- Voir les applications qui tourne sur le port 8080

```
netstat -ano | findstr :8080
```

- Arrêter le process

```
taskkill /PID 29648 /F
```

1.2 Exercice TT 2

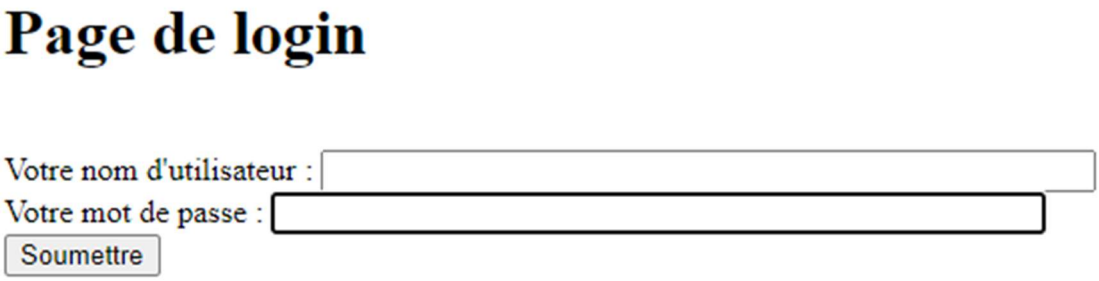
Durant l'exercice

1.3 Exercice TT 3

Durant l'exercice nous avons utilisé comme l'exercice précédent un index et un JSP. Cet exercice au contraire du deuxième nous pouvons interagir avec. Pour pouvoir interagir dans l'index, nous avons dû faire un post qui appelle le JSP.

```
<html>
    <head>
        <title>Login Html et check Jsp</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    </head>
    <body>
        <h1>Page de login </h1><br>
        <form method="post" action="checkLogin.jsp">  <!-- la page check-
Login.jsp sera appelée lors du submit -->
            <label for="user">Votre nom d'utilisateur :</label>
            <input type="text" name="username" id="user" size="50"
maxlength="50" /><br>
            <label for="pass">Votre mot de passe :</label>
            <input type="password" name="password" id="pass" size="50"
maxlength="50" /><br>
            <input type="submit" value="Soumettre">
        </form>
    </body>
</html>
```

Voici le rendu final de cet exercice, la première image c'est la page d'accueil du site est la 2ème page nous retourne les entrer que nous avons mis dans la page précédente.



1.4 Exercice TT 4

Cet exercice ressemble pas mal à l'exercice d'avant nous devons juste faire appel à une api.

Pour cela nous prenons de ce site l'exemple d'une ville.

```
<a href="https://www.prevision-meteo.ch/meteo/localite/fribourg-fr"><img
src="https://www.prevision-meteo.ch/uploads/widget/fribourg-fr_0.png"
width="650" height="250" /></a>
```

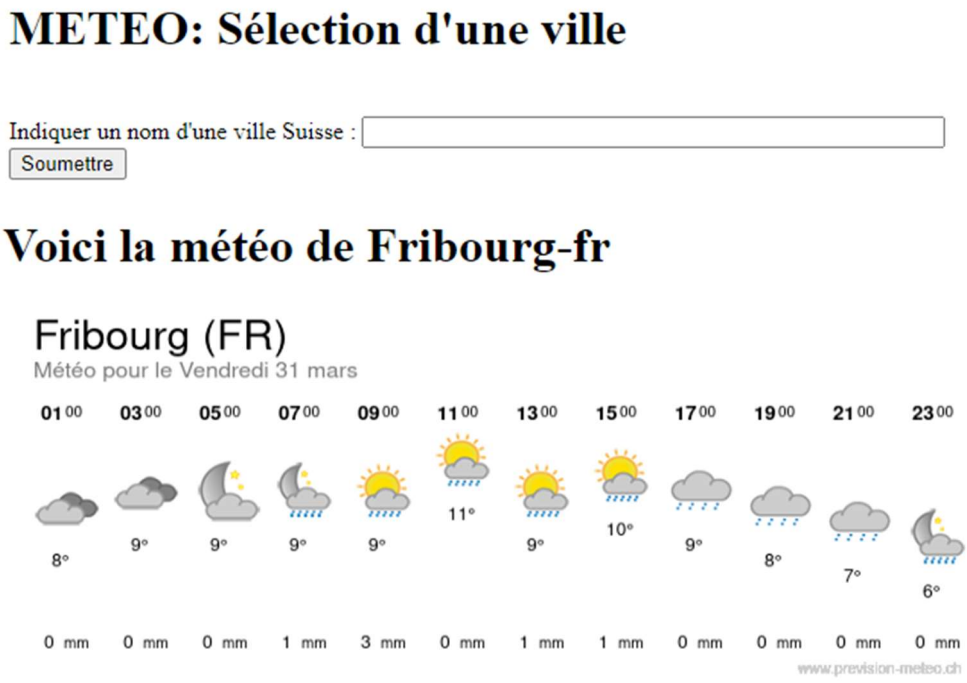
Mets à la place du « fribourg-fr » nous mettons ce que l'utilisateur à rentrer dans l'index.

```
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>JSP Page</title>
    </head>
    <body>
                <%
            String nomVille = request.getParameter("nomVille");


         %>
        <h1>Voici la météo de <%=nomVille%></h1>

<a href="https://www.prevision-meteo.ch/meteo/localite/<%=nomVille%>"><img
src="https://www.prevision-meteo.ch/uploads/widget/<%=nomVille%>_0.png"
width="650" height="250" /></a>
    </body>
</html>
```

Voici le rendu final, la première image c'est quand nous arrivons sur le site et la 2ème page c'est lorsque nous avons entré une ville valide pour l'api dans l'input.



1.5 Exercice TT 5

Durant cet exercice cette exercice nous avons afficher des villes qui était stocker dans une base de données. Pour afficher les villes nous devions faire de 2 manières une fois avec des « print » et la deuxième manière était de mettre les pays dans des « div ».

```
<%
    out.println("Solution avec instruction java out.println()");
    for (String pays : lstPays) {
        out.print("<div>");
        out.print(pays);
        out.print("</div>");
    }
%>
```

```
    <div>Solution avec intégration de variables java dans HTML</div>
        <%
            for(String pays : lstPays){
        %>
        <div><%=pays%></div>
        <%  } }   %>
```

1.6 Exercice TT 6

Dans cet exercice nous avons dû faire un login en vérifiant les entré inscrite par l'utilisateur. Si les entré était fausse il était redirigé vers une certaine page et s'il était juste vers une autre page.

Ci-dessous nous pouvons voir le jsp qui nous permet vérifier les donné et rediriger vers les bonnes pages.

```
<body>
   <% String nom = request.getParameter("nom");
   String prenom = request.getParameter("prenom");
   String motDePasse = request.getParameter("password");
   String remoteHost = request.getRemoteHost();
   String msg = "La connexion n'a pas réussi";
   if (nom.equals("Gambera") && motDePasse.equals("Pa$$w0rd") &&
prenom.equals("Luca") ) {
            Info.setNom(nom);
            Info.setPrenom(prenom);
            Info.setPassword(motDePasse);
    %>
    <jsp:forward page="maJspLogge.jsp"/>
    <%} else {
        Error.setHost(remoteHost);
        Error.setMsg(msg);
    %>
```

```
    <jsp:forward page="erreur.jsp"/>
    <%              }%>
</body>
```

Pour les beans info et Error il ne faut pas oublier d'implémenter « Serializable »

```
public class BeanError implements Serializable {
```

Pour la page html c'est comme les exercice précédent nous devons mettre le jsp vers le quelle il devra être redirigé.

```
<form method="post" action="maJspTraitement.jsp">
```

Pour la page jsp erreur et logged nous avons dû importer les beans comme le jsptraitement.

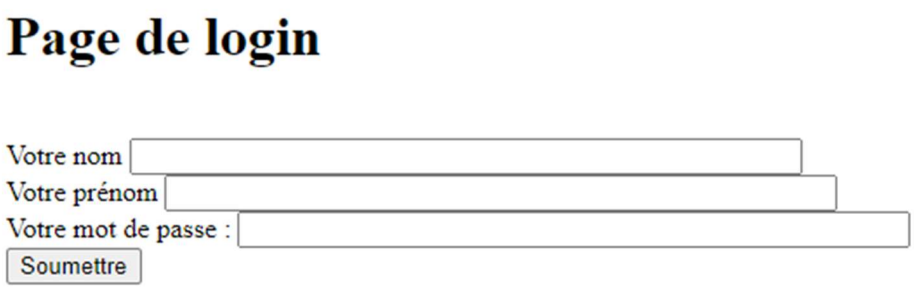
```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@ page import="beans.BeanInfo"%>
<jsp:useBean id="Info"  scope="session" class="beans.BeanInfo"/>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>JSP Page</title>
    </head>
    <body>
        <h1>Vous êtes loggé, voici les informations de l'utilisateur!</h1>
        <p>
            Nom: <%=Info.getNom()%> <br>
            Prenom: <%=Info.getPrenom()%>
        </p>
    </body>
</html>
```

Voici le jspErreur

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@ page import="beans.BeanInfo"%>
<%@ page import="beans.BeanError"%>
<jsp:useBean id="Error"  scope="session" class="beans.BeanError"/>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Erreur</title>
    </head>
    <body>
        <h1>Page d'erreur</h1>
        <p>
            Accès non autorisé pour hôte <%=Error.getHost()%>
```

```
        </p>
        <a href="index.html">
            Retour à la page de login
        </a>
    </body>
</html>
```

Cette image est lorsque vous arrivé sur le site.



Cette image est lorsque que vous avez inscrit les bonne entré.

Cette image est lorsque que vous avez inscrit les mauvais entrés.

1.7 Exercice TT 7

Dans cet exercice nous avons dû utiliser un Servlet. Le Servlet nous permet de mettre en place en dynamiquement des données dans un serveur web. Les données sont en HTML.

Nous avons repris l'exercice 3, et nous avons changé l'action du form pour pouvoir mettre le Servlet.

```
<form method="post" action="MaServlet">
```

Après avoir fait cela nous devons implémenter la méthode « processRequest » du Servlet. >Vous pouvez le voir ci-dessous.

```
protected void processRequest(HttpServletRequest request, HttpServletRes-
ponse response) throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    try ( PrintWriter out = response.getWriter()) {
        /* TODO output your page here. You may use following sample code. */
        out.println("<!DOCTYPE html>");
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Servlet MaServlet</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>Servlet Servlet at " + request.getContextPath() +
"</h1>");
        out.println("<p>Votre nom d'utilisateur est : " + request.getParame-
ter("username") + " et votre mot de passe est : " + request.getParame-
ter("password"));
        out.println("</body>");
        out.println("</html>");
    }
}
```

Voici la page d'accueil du site :

Voici le résultat quand nous appuyons sur le bouton soumettre.

1.8 Exercice TT 8

Dans cet exercice nous avons dû faire comme l'exercice 6, checker les entrer données mais dans cet exercice nous devons le faire avec un Servlet. Il y a comme l'exercice 3 une page html avec un formulaire et des entrés à inscrire.

Dans le Servlet nous allons utiliser la méthode « doPost ». En premier nous allons récupérer la session et paramétrer le temps d'inactivation, puis nous récupérons les données (username, password). Si les crédenciale sont juste nous allons mettre le nouveau nom et son attribut dans la session. Et pour finir nous redirigeons vers la page logged.

Si ce n'est pas les bon crédenciales nous allons charger la page erreur

Ci-dessous vous retrouver le code dons nous venons de parler.

```
protected void doPost(HttpServletRequest request, HttpServletResponse re-
sponse) throws ServletException, IOException {
    try ( PrintWriter out = response.getWriter()) {
        HttpSession session = request.getSession();
        session.setMaxInactiveInterval(20);
        String username = request.getParameter("username");
        String password = request.getParameter("password");
        RequestDispatcher dispatch;
        if (username.equals("admin") && password.equals("Pa$$w0rd")) {
            BeanInfo info = new BeanInfo();
            info.setNom(username);
            info.setPassword(password);
            session.setAttribute("Info", info);
            dispatch = request.getRequestDispatcher("pageAutorise.jsp");
        } else {
            BeanError err = new BeanError();
            err.setHost(request.getRemoteHost());
            err.setMsg("mauvais identifiant");
            session.setAttribute("Error", err);
            dispatch = request.getRequestDispatcher("erreur.jsp");
        }
        dispatch.forward(request, response);
    }
}
```

Voici le code de si non avons réussi à nous logger

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@ page import="beans.BeanInfo"%>
<%@ page import="beans.BeanError"%>
<jsp:useBean id="Info"  scope="session" class="beans.BeanInfo"/>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>JSP Page</title>
    </head>
    <body>
        <h1>Vous êtes logguer</h1>
        <p>
            Votre nom est <%=Info.getNom()%> <br>
            Votre password est <%=Info.getPassword()%><br>
        </p>
    </body>
</html>
```

Voici le code erreur

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@page import="beans.BeanInfo"%>
<%@page import="beans.BeanError"%>
<jsp:useBean id="Error"  scope="session" class="beans.BeanError"/>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>JSP Page</title>
    </head>
    <body>
        <h1>Vous n'êtes pas logguer</h1>
        <p>
            Votre Host est <%=Error.getHost()%> <br>
            Votre message est <%=Error.getMsg()%><br>
        </p>
    </body>
</html>
```

Voici la page d'accueil :

Page de login

Votre nom d'utilisateur : Votre mot de passe : Soumettre

La page si vous est logguer

Vous êtes logguer

Votre nom est admin
Votre password est Pa$$w0rd

La page d'erreur

Vous n'êtes pas logguer

Votre Host est 127.0.0.1
Votre message est mauvais identifiant

1.9 Exercice TT 10

Durant cet exercice nous avons dû faire une application client et serveur pour accéder à des données qui sont stocker sur le serveur

1.9.1 Client

Pour le client nous allons commencer à faire 2 boutons qui va nous permettre de faire une demande de récupérer l'auteur et le message.

```
<form method="post" action="ServletCtrl">
    <input type="submit" name="getMessage" value="Récupérer message"/>
    <input type="submit" name="getAuthor" value="Récupérer auteur"/>
</form>
```

Ensuite nous faisons 2 classes dont une classe est un servlet. La premier classe ClientMessage va nous permettre de faire la communication avec le serveur.

```
public class ClientMessage {


    private WebTarget webTarget;
    private Client client;
    private static final String BASE_URI = "http://gamberal01.emf-
informatique.ch/javaSimple_Rest_Server/webresources";


    public ClientMessage() {
        client = javax.ws.rs.client.ClientBuilder.newClient();
        webTarget = client.target(BASE_URI).path("tutoriel");
    }


    public String getAuthor() throws ClientErrorException {
        WebTarget resource = webTarget;
        resource = resource.path("getAuthor");
        return
resource.request(javax.ws.rs.core.MediaType.TEXT_PLAIN).get(String.class);
    }


    public String getMessage() throws ClientErrorException {
        WebTarget resource = webTarget;
        resource = resource.path("getMessage");
        return
resource.request(javax.ws.rs.core.MediaType.TEXT_PLAIN).get(String.class);
    }


    public void close() {
        client.close();
    }


}
```

La classe ServletCtrl on a dû faire des ifs pour savoir si nous voulions recevoir le message, l'auteur ou s'il y a une erreur.

```
protected void processRequest(HttpServletRequest request,
HttpServletResponse response) throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    try ( PrintWriter out = response.getWriter()) {
        /* TODO output your page here. You may use following sample code. */
        if (request.getParameter("getMessage")!=null){
            String reponse = client.getMessage();
            out.println(reponse);

        }

        else if (request.getParameter("getAuthor") != null){
            String reponse = client.getAuthor();
            out.println(reponse);
        }
        else {
            out.println("Quelque chose ne s'est pas bien passé !");
        }
    }
}
```

1.9.2 Serveur

Dans le serveur nous avons dû aussi créer deux classes, une qu'on va appeler « ApplicationConfig » avec son path « webresources ». Ce path va nous permettre de rediriger vers les bonnes ressources.

```
@javax.ws.rs.ApplicationPath("webresources")
public class ApplicationConfig extends Application {
```

Pour la classe Message nous avons aussi dû mettre un path « tutoriel » et ensuite pour chaque méthode nous avons aussi dû mettre un path. Si nous mettons « /webresources/tutoriel/ getMessage » cela va nous donner « Bonjour tout le monde »

```
@Path("tutoriel")
public class Message {
    @Context
    private UriInfo context;
    public Message() {
    }
    @GET
    @Path("getMessage")
    @Produces(javax.ws.rs.core.MediaType.TEXT_PLAIN)
    @Consumes(javax.ws.rs.core.MediaType.APPLICATION_FORM_URLENCODED)
    public String getMessage() {
        return "Bonjour tout le monde !";
```

```
    }
    @GET
    @Path("getAuthor")
    @Produces(javax.ws.rs.core.MediaType.TEXT_PLAIN)
    @Consumes(javax.ws.rs.core.MediaType.APPLICATION_FORM_URLENCODED)
    public String getAuthor() {
        return "Fait par Gambera Luca !";
    }
```