Exercices sur Servlets/JSP

Installation

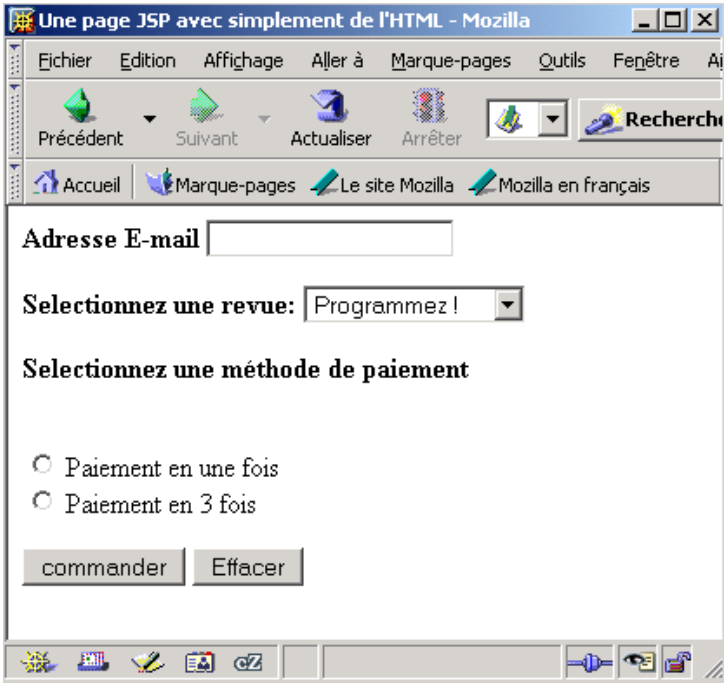
- Installer, si ce n'est déjà fait :
  - J2SE (1.6)
  - tomcat (6.0)
  - un éditeur de texte (notepad++)

Correspondance URL vs. emplacement disque

- On doit avoir la correspondance :
  `http://localhost:8080/`
  <=>
  `%TomcatHome%\webapps\ROOT\`
- Voir (ou construire) des démos de
  `http://localhost:8080/`
  `http://localhost:8080/test2JMF.html`
  `http://localhost:8080/test2JMF.jsp`

Exercice 1 (1/3)

- Ecrire la page html suivante :



- Le menu est composé de Java SkyLine, Java World, Programmez ! (menu par défaut), Devellopez.com
- La transformer en page jsp. Qu'y a t il à faire ?

Déploiement d'une application web

- Remarque 1
  Toute application web (= site web mis `%TOMCAT_HOME%\webapps\MonAppliWeb`) sera accessible par des URL commençant par `http://localhost:`*`port`*`/MonAppliWeb`
- Remarque 2
  Toute application web (= site web mis `%TOMCAT_HOME%\webapps\MonAppliWeb`) est décrite par son fichier `%TOMCAT_HOME%\webapps\MonAppliWeb\WEB-INF\web.xml`

Déploiement d'une servlet

- Mettre dans le `web.xml` de votre application web, les lignes :

```
<servlet>
    <servlet-name>unNomQuelconque</servlet-name>
    <servlet-class>AfficheChoixDansLaJSPServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>unNomQuelconque</servlet-name>
    <url-pattern>/AfficheChoixDansLaJSPServlet</url-pattern>
</servlet-mapping>
```

- Remarque 3
  Les servlets de l'application web doivent être rangées dans `MonAppliWeb\WEB-INF\classes`

Exercice 1 (2/3)

- Placer la page jsp dans l'arborescence tomcat (par exemple sous `TomcatHome\webapps\ROOT\`) et accéder à cette jsp par un navigateur à une URL commençant par `http://localhost ...`
- Lorsqu'on clique sur le bouton commander de la page jsp précédente, on demande à lancer la servlet d'URL `/AffichеChoixDansLaJSPServlet`
- Accéder à cette page JSP par le navigateur (lancer ce qu'il faut pour cela)
- Indiquer où se trouve la servlet générée associée à cette JSP (traduction de cette JSP).

Exercice 1 (3/3)

- Ecrire la servlet `AfficheChoixDansLaJSPServlet`. Cette servlet affiche les choix qui ont été fait par l'utilisateur dans la page JSP précédente.

Sessions

- Une session = Une suite d'interactions entre un client et un serveur Web
  - Elle couvre plusieurs requêtes HTTP sur une période donnée
- On peut utiliser les sessions pour
  - Mémoriser les actions d'un utilisateur unique
  - Exemples
    - achats en ligne (panier de commande)
    - Examens à distance
- HTTP : un protocole sans état => Il faut une technique pour mémoriser les sessions

HttpSession

- L'API Servlet fournit l'interface `HttpSession`
  - Les objets **`HttpSession`** mémorisent des données pendant une suite d'interactions d'un utilisateur (utilisant le même navigateur) sur l'application web
  - Fonctionne comme une table de hachage,
  - Est stocké coté serveur
  - Repéré par un sessionId échangé entre le serveur web et le client web

Utiliser `HttpSession` (1/2)

- Dans un premier temps on récupère la session (objet de la classe `HttpSession`) par
  `HttpSession getSession()`
  lancé sur `request`
- Cet appel retourne la session courante associée à la requête. Si la session n'existe pas, elle sera créée.
- Une fois la session obtenue on peut mémoriser des données par
  `void setAttribute(String name, Object value)`ou les récupérer par :
  `Object getAttribute(String name)`

Utiliser `HttpSession` (2/2)

```
HttpSession session = request.getSession();
ArrayList laListe = ...
… … …
session.setAttribute("cnam", laListe);
… … …
ArrayList recupListe =(ArrayList)
   session.getAttribute("cnam");
```

Exercice 2 (1/2)

- Construire une servlet qui affiche un compteur qui est incrémenté à chaque accès sur cette servlet pendant une session.
- Indication : on pourra utiliser la classe `Integer` qui modélise le type `int`.
- Remarque importante : il faut que votre navigateur ait activé les cookies

Gestion du session id (1/2)

- Par défaut le session id est passé comme cookie du serveur au client web (qui le lui renvoie à chaque requête cf. gestion des cookies dans http)
- A condition que le client web autorise les cookies
- Si ce n'est pas le cas, il faut passer le session id de sorte à le récupérer lors de la prochaine interaction
- => la solution. Le mettre dans tous les liens de la page retournée car sinon …
- compléter !!

Gestion du session id (2/2)

- La technique :
- Utiliser sur `response` la méthode `encodeURL()` ayant pour argument l'URL à encoder
- exemple :

```
String leLienEncode =
   response.encodeURL("/servlet/CounterServlet");
out.println("<a href=" + leLienEncode + ">cliquez ici</a>");
```

- il faut faire cela sur toutes les liens de la page retournée.
- Pourquoi ?

Exercice 2 (2/2)

- Relancer l'exercice précédent en déactivant les cookies sur votre navigateur
- Réécrire votre servlet de sorte que le compteur soit incrémenté même sans l'utilisation des cookies

Exercice 3

- Ecrire une architecture MVC coté serveur. On pourra utilisé la page JSP du premier exercice.