TP 1
Introduction aux Servlets

Resum

Dans ce TP vous allez deployer vos premiers servlets et decouvrir la structure d'un conteneur de servlet (Tomcat). Pour le moment, pas d'IDE (ex : Eclipse), on code a l'ancienne. Les seules choses dont vous avez besoin pour realiser les t^aches demandees sont un terminal et un navigateur web (on peut aussi lancer un navigateur web dans un terminal (Lynx par exemple)).

Q1.1 Installation de Tomcat

{ Assurez vous que le SDK Java est bien installe sur votre VM, en lancant la commande javac -version. Si ce n'est pas le cas, installez java avec la commande sudo apt-get install openjdk-7-jdk

{ Telechargez Tomcat:

{ Editez le chier conf/server.xml pour que Tomcat utilise le port 80 au lieu de 8080.

{ Demarrez le serveur en executant le script bin/startup.sh en tant que super-utilisateur (seuls root ou le super-utilisateur peuvent demarrer un serveur utilisant le port 80)

{ Les messages d'erreurs de Tomcat sont enregistres dans le chier logs/catalina.out , vous pouvez veri er ici que le serveur s'est lance sans erreur.

{ Pour veri er que le serveur est bien lance, utilisez la commande lynx http://localhost, la page d'accueil de Tomcat devrait s'a cher.

{ Veri ez que le serveur est bien accessible depuis votre machine, dans la barre d'adresse de votre navigateur, l'adresse : http://127.0.0.1:80

{ Vous pouvez stopper le serveur avec le script bin/shutdown.sh.

Q1.2 Votre premier Servlet : Hello World !

Il faut tout d'abord creer l'arborescence des dossiers : a la racine du serveur Tomcat se trouve un dossier webapps. Dans ce dossier, creez un repertoire hello. Les chiers de ce repertoire seront accessibles via http://fvotre-loging.rmorpheus.enseirb.fr/hello.

Les di erentes applications presentes dans le serveur Tomcat se trouvent dans le dossier we-bapps. C'est ici que nous allons creer notre premier servlet. Dans ce dossier, creez un repertoire hello.

Dans le dossier hello, creez un dossier WEB-INF. C'est ici que seront situees les informations liees a la con guration de l'application.

Dans le dossier WEB-INF, ajoutez un dossier classes. Comme son nom l'indique, c'est ici que devront ^etre deployees les di erentes classes utilisees par l'application.

Creez en n un dossier src dans le dossier webapps/hello, puis ajoutez dans ce dossier un nouveau chier HelloServlet.java, contenant le code de la gure 1.

```
package fr.enseirb.t2.servlet;

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

@WebServlet(urlPatterns="/world")
public class HelloServlet extends HttpServlet
{
   @Override
   protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws
      ServletException, IOException {

      resp.setContentType("text/html");
      PrintWriter out = resp.getWriter();
      out.println("<html>");
      out.println("<body>"); out.println("Hello
      World!"); out.println("</body>");
      out.println("</html>");

      }
}
```

Figure 1 { HelloServlet.java

Vous pouvez ensuite compiler la classe HelloServlet avec la commande suivante.

```
javac -classpath ../../../lib/servlet-api.jar HelloServlet.java
```

Deplacez le chier .class obtenu dans WEB-INF/classes/fr/enseirb/t2/servlet/. Notez que la n de l'arborescence correspond au nom du package de HelloServlet.

NB : mkdir -p permet de creer une arborescence de chiers en une seule commande.

Redemarrez votre serveur (shutdown.sh puis startup.sh).

En tapant l'adresse adresse-du-serveur/hello/world, le serveur fera un appel a la methode doGet, et a chera une page html avec le contenu ajoute par le PrintWriter.

Q1.3 Traitement des parametres d'une requ^ete GET

Il est possible de passer des parametres dans une requ^ete HTTP GET. L'url prend alors la forme suivante : http://localhost:8080/hello/world?name=toto

Modi ez la classe HelloServlet a n que l'url ci-dessus a che une page html contenant le texte : "Hello toto."

Pour recuperer la valeur d'un parametre, il faut utiliser :

```
String parameterValue = req.getParameter("parameterName");
```

Pour eviter de redemarrer le serveur a chaque fois qu'on fait une modi cation des chiers .class, une solution est d'editer lechier conf/context.xml et d'ajouter un attribut a la balise Context :

```
<Context reloadable="true">
....
```

Redemarrez le serveur.

Lorsque vous ferez des modi cations des chiers .class, le message suivant devrait s'a cher dans les logs de Tomcat : INFO: Reloading Context with name [/hello] is completed.

Q1.4 Requ^etes POST

Un servlet peut egalement traiter des requ^etes POST. Ce type de requ^ete peut ^etre envoy via un formulaire html. Ajoutez dans le dossier webapps/hello le chier index.html suivant :

```
<html><body>
  <form action="world" method=POST>
    <p>Please Fill the Registration Form</p><br> Enter Your
    Name<input type="text" name="name"><br> <input
    type="submit" value="send">
  </form>
</body></html>
```

Listing 1 { index.html

Si vous ouvrez la page http://adresse-du-serveur/hello, vous verrez appara^tre le formulaire. Si vous cliquez sur le bouton send, une exception HTTP sera renvoyee par le serveur, signi ant que la methode POST n'est pas supportee par le serveur.

Il faut donc ajouter le support de la methode POST a notre servlet. Ajoutez dans le chier HelloWorld.java la methode doPost :

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

  resp.setContentType("text/plain");
  // TODO: Traiter les parametres de la requete

}
```

Notez ici que le type de reponse est text/plain, ce qui signi e qu'il est possible de mettre du simple texte dans la reponse, au lieu de texte formate en Html.
Modi ez le code pour a cher "Hello toto !" lorsque "toto" est entr dans le formulaire.

Q1.5 Headers

Un certain nombre d'informations sont disponibles dans les headers des requ^etes, comme le nom de l'h^ote, le navigateur du client, etc. La fonction suivante permet d'a cher les headers d'une requ^ete dans une table html. Copiez-la dans votre servlet, puis testez-la avec un appel a cette fonction dans la methode doGet().

```
  private void printHeaders(HttpServletRequest req, PrintWriter out) {
    out.println("<br><b>Request Method: </b>" +
      req.getMethod() + "<br>\n" +
      "<b>Request URI: </b>" + req.getRequestURI() + "<br>\n" + "<b>Request
      Protocol: </b>" + req.getProtocol() + "<br><br>\n" + "<table border=1
      align=center>\n" + "<tr bgcolor=\"#FFAD00\">\n" + "<th>Header
      Name<th>Header Value");
    Enumeration headerNames = req.getHeaderNames();
    while(headerNames.hasMoreElements()) {
      String headerName = (String)headerNames.nextElement();
      out.println("<tr><td>" + headerName);
      out.println("          <td>" + req.getHeader(headerName));
```

```
    }
    out.println("</table>");
  }
```

Listing 2 { A chage des headers

Pensez a ajouter l'import suivant :

```
import java.util.Enumeration;
```

Faites la m^eme chose pour la methode doPost (pensez a modi er le content-type de la reponse en text/html).

Vous remarquerez qu'un nouveau header content-length est apparu. Vous pouvez (et devez) donc lire le contenu de la requ^ete et l'a cher gr^ace a l'instruction suivante.

```
String content = req.getReader().readLine();
```

Q1.6 Sessions

Il est possible de conserver des donnees le temps d'une session. Une session se termine lorsque le timeout expire ou lorsque le serveur s'arr^ete. Modi ez votre servlet pour que le nom de la derniere personne soit a che si aucun nom n'est mis en parametre.

La session se comporte comme un table de hachage, associant des cles de type String a des elements de type Object.

Pour enregistrer un element dans la session, utilisez la methode :

```
req.getSession().setAttribute("name", userName);
```

Et pour recuperer la valeur d'un element, utilisez :

```
String userName = (String) req.getSession().getAttribute("name");
```

Q1.7 Forward

Di erentes parties de l'application peuvent traiter une m^eme requ^ete. Cette operation s'appelle le forwarding, et elle est transparente du point de vue du client. Le forwarding se fait gr^ace a la methode suivante : req.getRequestDispatcher("/servletToto").forward(req, resp);
Creez un servlet qui forward les requ^etes GET et/ou POST vers un servlet di erent.

Q1.8 Redirection

Il est possible de rediriger le navigateur du client vers une autre URL, qui peut ^etre externe au serveur (e.g. www.google.fr), grace a la methode sendRedirect de la classe HttpServletResponse. Creez un formulaire qui redirige le navigateur du client sur Google, en ayant lance la recherche avec les mots cles contenus dans le champ de saisie.