

Java. Les servlets

Jean-Pierre Fournier, 2006

<http://www.iut-orsay.fr/~fournier>

Une application client/serveur

- ☐ possède des éléments qui fonctionnent côté client
 - un navigateur présentant des pages Web
 - ☐ statiques
 - ☐ dynamiques, comportant par exemple des applets Java
 - une application, notamment une application Java
- ☐ possède des éléments qui fonctionnent côté serveur
 - cgi, php, asp, jsp...
 - servlets Java

Une applet Java

- ❑ est téléchargée depuis le site Web vers le poste client quand la page Web qui la référence est appelée
- ❑ est exécutée par la machine virtuelle Java du poste client
- ❑ c'est la machine virtuelle qui a été associée à l'explorateur
- ❑ mais c'est aussi un Panel qui peut être inclus dans une interface graphique d'application

Une servlet Java

- ❑ n'est présente a priori que sur le serveur
- ❑ s'exécute avec la machine virtuelle Java du serveur
- ❑ répond aux requêtes envoyées par les clients
- ❑ possède toutes les qualités des applicatifs développés avec Java
 - qualité du code, portabilité, langage fortement typé, langage « objet » : héritage, généricité, encapsulation, etc.

Apache et Tomcat

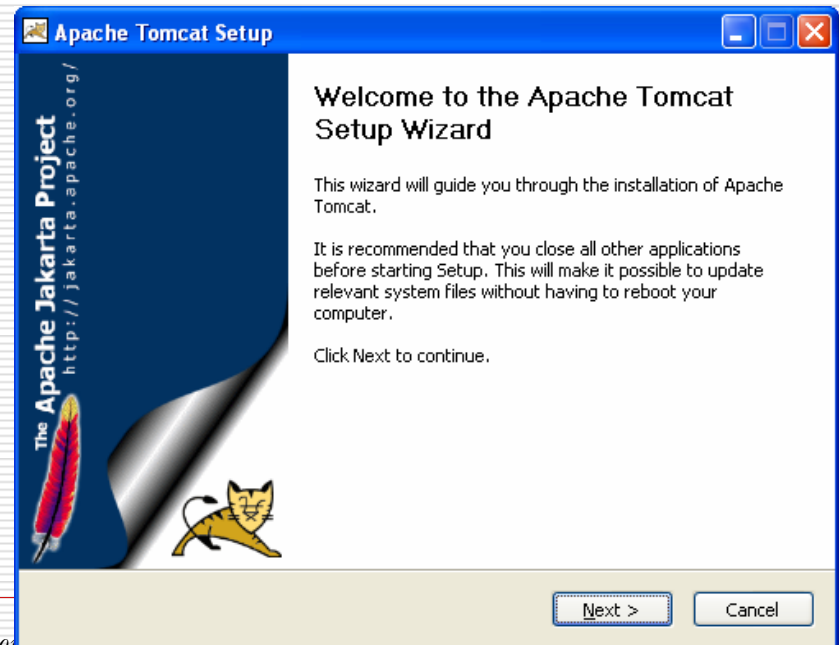
- Apache (<http://www.apache.org>) est un des serveurs Web les plus employés : il est capable de recevoir des requêtes http et de leur répondre
- Tomcat (<http://tomcat.apache.org>) est le complément d'Apache qui permet à ce serveur Web de déclencher des éléments de code Java, donc des servlets

Un exemple complet

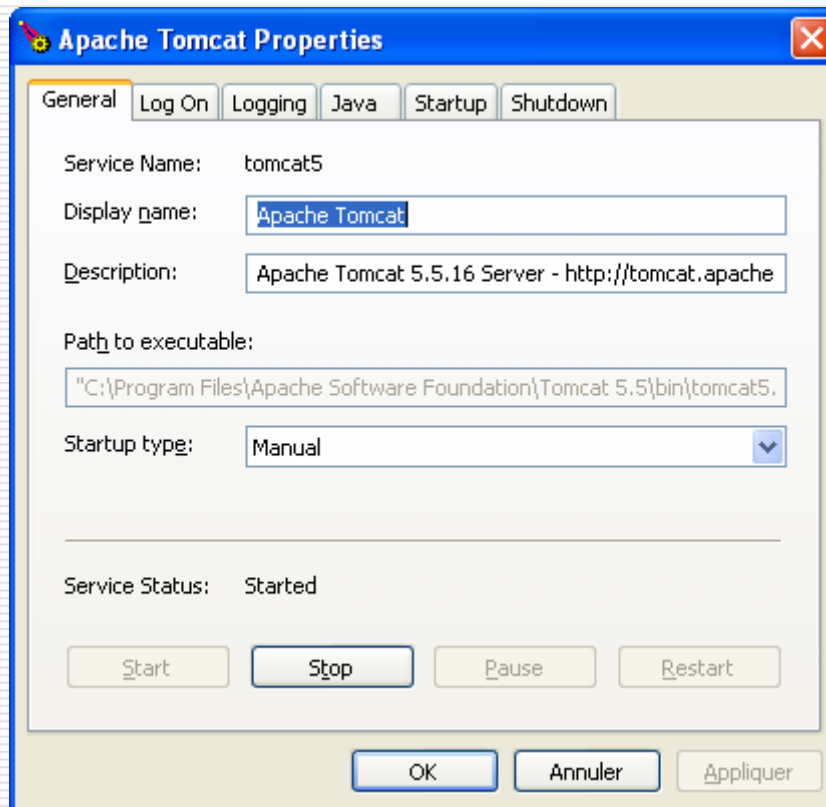
- ❑ Nous allons construire une application client/serveur complète, employant Java côté client (applet et application) et côté serveur (servlet), en donnant le détail des étapes successives, de l'installation des outils aux tests de bon fonctionnement
- ❑ Dans cette application, un serveur dispose d'informations sur des animaux et les bruits qu'ils produisent. Le client peut donner le nom d'un animal et recevoir le bruit associé

Etape 1 : installation et configuration de Tomcat

- Téléchargement depuis <http://tomcat.apache.org>
 - sous Windows, le plus simple est de choisir « Windows Service Installer »
 - Le lanceur de tomcat est le programme tomcat6w.exe (en version 6)



Mise en route de Tomcat



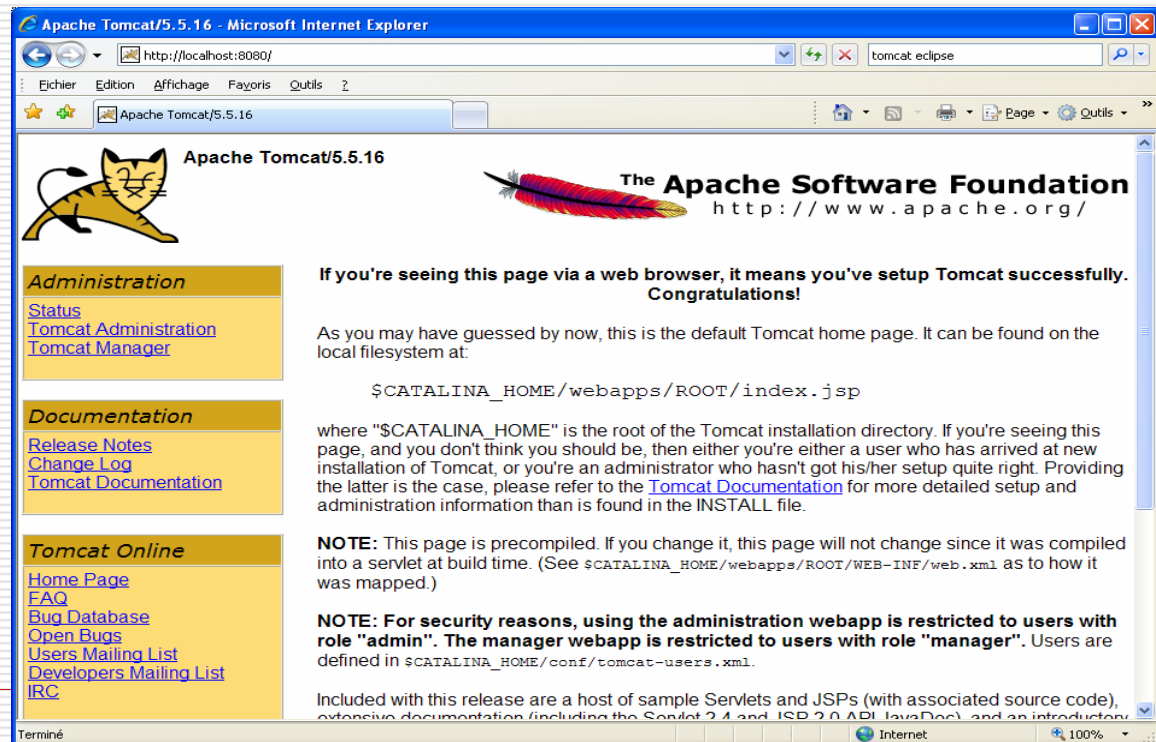
Etape 2 : installation du plugin Tomcat pour Eclipse

- ❑ S'installe depuis <http://www.sysdeo.com>
- ❑ le fichier téléchargé doit être directement décompressé dans le répertoire plugins, là où eclipse a été installé
- ❑ Au lancement suivant d'eclipse, 3 nouveaux icônes apparaissent, permettant de démarrer, arrêter et redémarrer Tomcat sans avoir à passer par l'interface standard, les messages parvenant directement dans la console eclipse...



Vérification de l'installation

- Si tout s'est bien passé, en tapant <http://localhost:8080/> dans le navigateur :



Etape 3 : Mise en place de notre servlet

- Cette servlet doit répondre aux requêtes « get » et « post »

```
public class MaServlet extends
    HttpServlet {

private String[] questions =
    {"chien", "chat", "ane"};

private String[] reponses =
    {"ouaf", "miaou", "hi-han"};
```

La méthode doGet

@Override

```
protected void doGet(HttpServletRequest arg0,  
    HttpServletResponse arg1) throws ServletException,  
    IOException {  
    System.err.println("methode get appelée maintenant");  
    arg1.setContentType("text/html");  
  
    PrintWriter out = arg1.getWriter();  
  
    out.println("<html><head></head><body>");  
    out.println("Je réponds !!!");  
    out.println("</body></html>");  
}
```

La méthode doPost

@Override

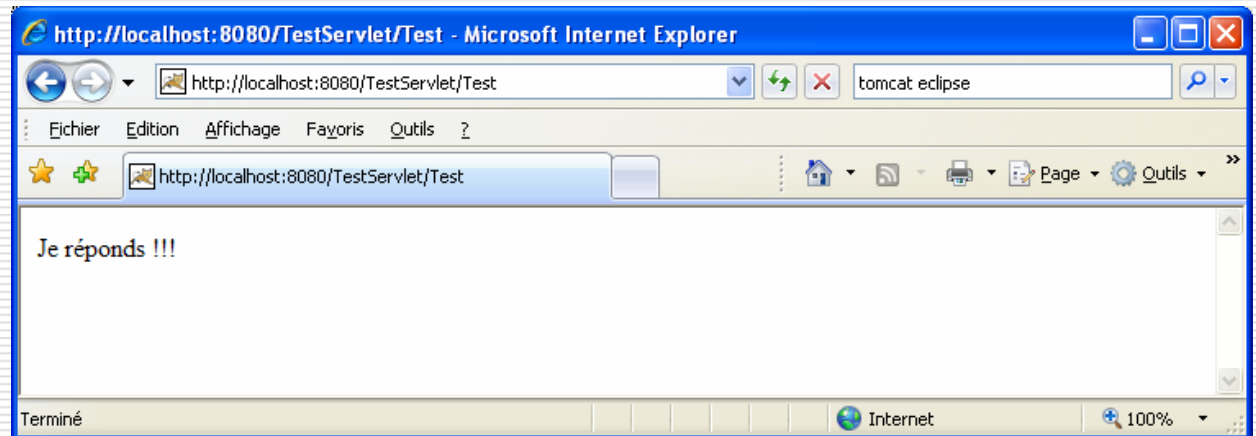
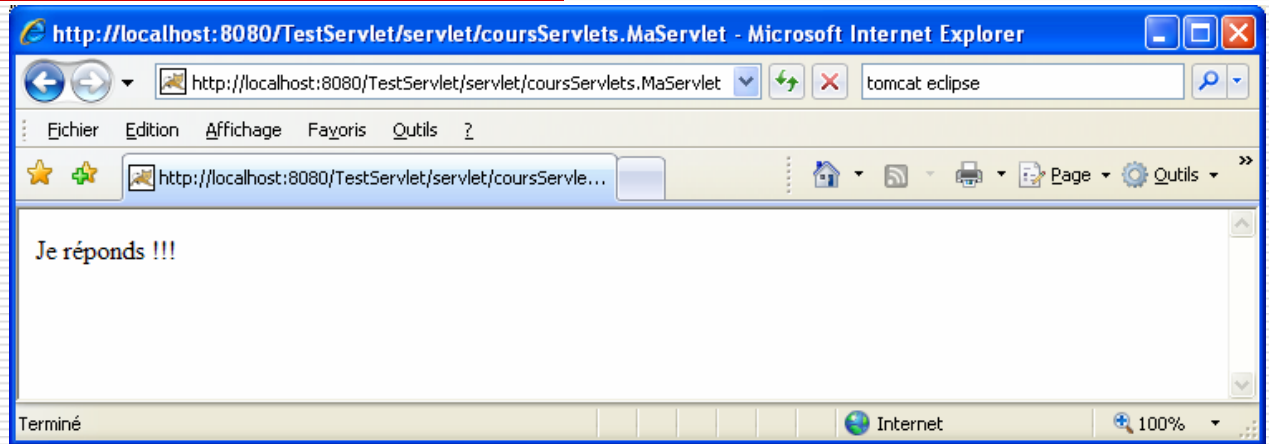
```
protected void doPost (HttpServletRequest arg0,  
    HttpServletResponse arg1) throws  
    ServletException, IOException {  
  
    arg1.setContentType("text");  
    PrintWriter out = arg1.getWriter();  
    out.println(  
        recherche(arg0.getParameter("nom")) );  
}
```

La fonction de recherche

```
private String recherche(String parameter) {  
    if (parameter.equals("*")) {  
        String res = "";  
        for (String s : reponses)  
            res += s + ", ";  
        return res;  
    }  
    for (int i=0; i<questions.length; i++)  
        if (parameter.equals(questions[i]))  
            return reponses[i];  
    return "pas trouvé";  
}
```

Vérification

- ☐ lien direct
- ☐ lien caché



Pour que le lien direct fonctionne

- Dans le fichier de configuration de tomcat : web.xml, décommenter :

```
<!-- The mapping for the invoker  
servlet -->
```

```
<servlet-mapping>
```

```
  <servlet-name>invoker</servlet-name>
```

```
  <url-pattern>/servlet/*</url-pattern>
```

```
</servlet-mapping>
```

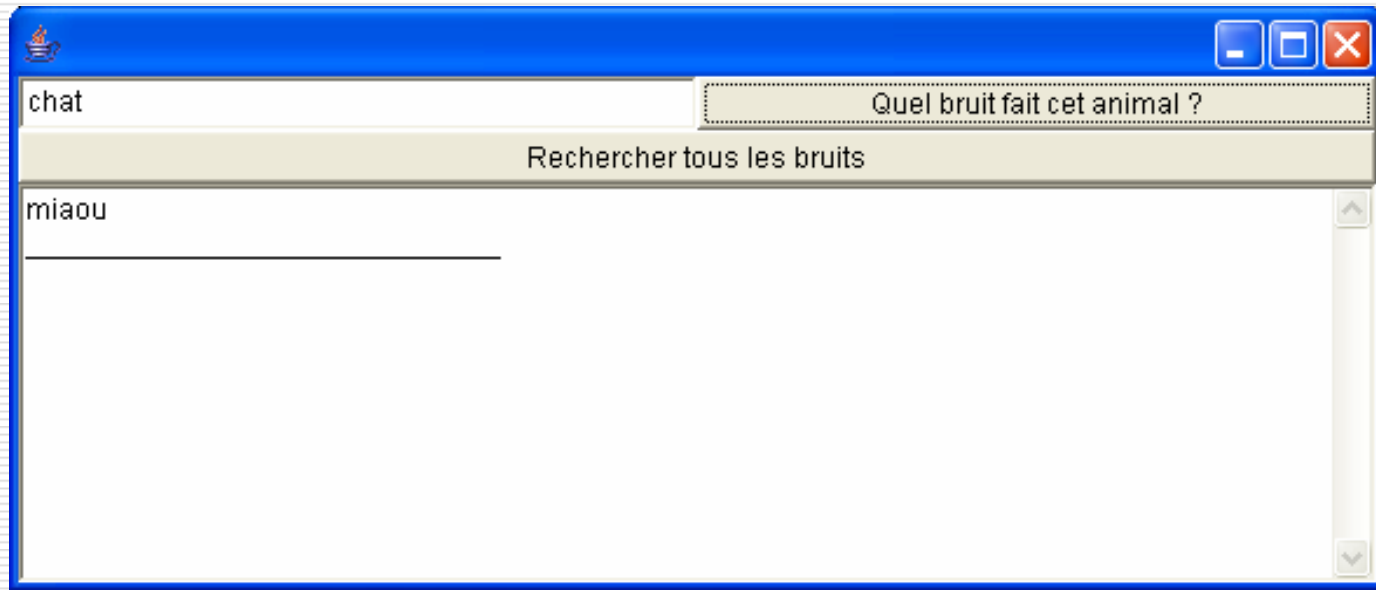

Pour que le lien « caché » fonctionne

- ❑ Dans le projet eclipse, dans le répertoire WEB-INF, créer un fichier web.xml contenant :

```
<?xml version="1.0" encoding="UTF-8"?>
<!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>TestServlet</display-name>
<servlet>
    <servlet-name>Test</servlet-name>
    <servlet-class>coursServlets.MaServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>Test</servlet-name>
    <url-pattern>/Test</url-pattern>
</servlet-mapping>
</web-app>
```

Etape 4. Déclencher la servlet depuis une application Java sur le client

- ❑ Voici à quoi ressemble l'interface graphique de notre application



Le code java (init)

```
public class DeclencheServlet extends Applet {
    private TextField nomCherche;
    private Button rechercheIsolee, rechercheTotale;
    private TextArea affichage;

    @Override
    public void init() {
        super.init();
        nomCherche = new TextField(10);
        rechercheIsolee = new Button("Quel bruit fait cet animal ?");
        rechercheTotale = new Button("Rechercher tous les bruits");
        affichage = new TextArea(10, 80);
        Panel haut = new Panel();
        haut.add(nomCherche); haut.add(rechercheIsolee);
        haut.setLayout(new GridLayout(1, 2));
        setLayout(new BorderLayout()); add(haut, BorderLayout.NORTH);
        add(rechercheTotale, BorderLayout.CENTER); add(affichage, BorderLayout.SOUTH);
    }
}
```

Le code Java (start)

```
public void start() {
    super.start();
    rechercheIsolee.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent arg0) {
            System.out.println("action simple");
            if (nomCherche.getText().equals("")) return;
            OutputStreamWriter writer = null;
            BufferedReader reader = null;

            try {
                // encodage des paramètres de la requête
                String donnees = URLEncoder.encode("nom", "UTF-8") +
                    "=" + URLEncoder.encode(nomCherche.getText(), "UTF-8");
                // création de la connection
                URL url = new URL("http://localhost:8080/TestServlet/Test");
                URLConnection conn = url.openConnection();
                conn.setDoOutput(true);
```

Code de start (2)

```
// envoi de la requête
writer = new OutputStreamWriter(conn.getOutputStream());
writer.write(donnees);
writer.flush();
// lecture de la réponse
reader = new BufferedReader(
    new InputStreamReader(conn.getInputStream()));
String ligne;
while ((ligne = reader.readLine()) != null) {
    affichage.append(ligne);
    affichage.append(
        new String("\n_____ \n"));
}
} catch (Exception e) { /* */}
finally {try {writer.close();
} catch (Exception e) { /* */}
try {
if (reader!=null) reader.close();
} catch (Exception e) { /* */}}}});
```

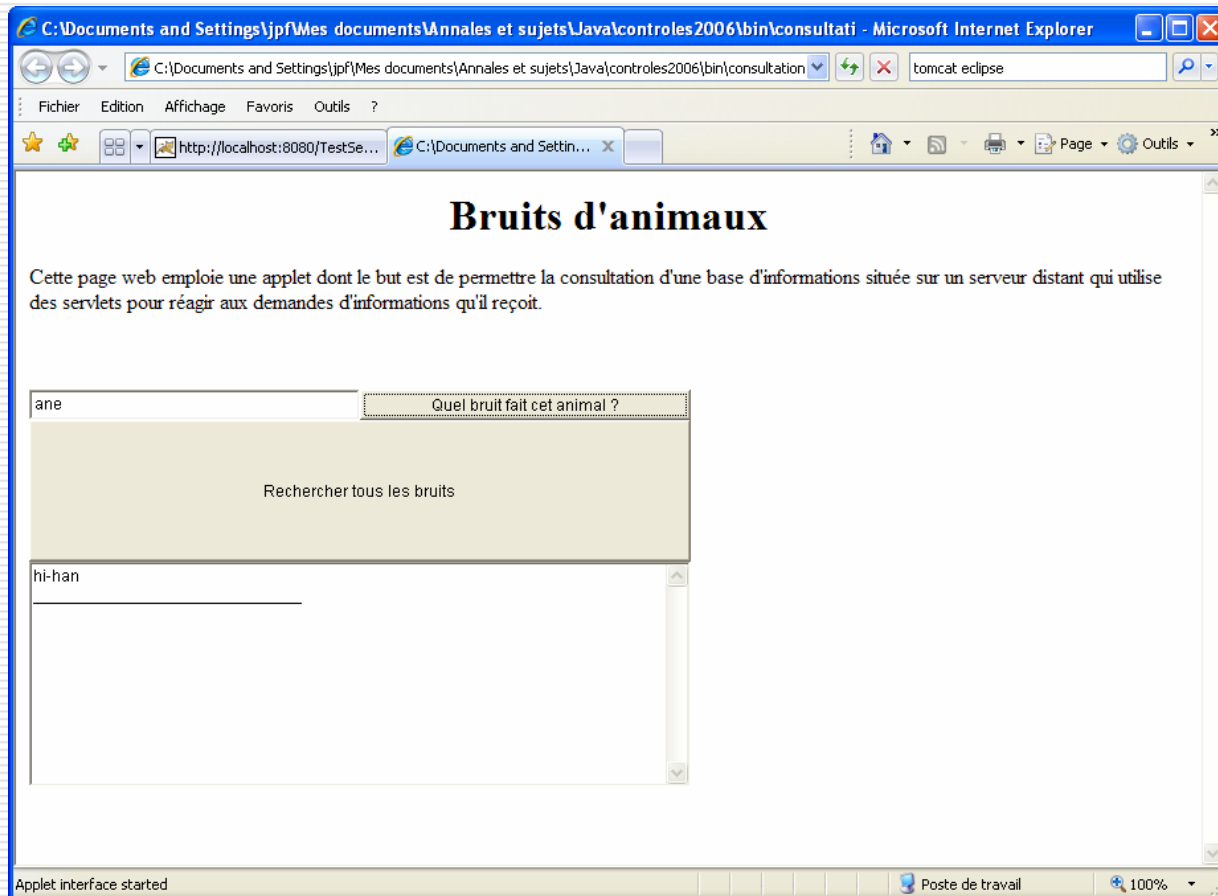
Traitement associé au second bouton

```
rechercheTotale.addActionListener(new
    ActionListener() {
        public void actionPerformed(ActionEvent e) {
            System.err.println("action complexe");
// pour utiliser le traitement de l'autre
// actionListener, produisons un événement
// virtuel...
            nomCherche.setText("*");
            rechercheIsolee.dispatchEvent(
                new(ActionEvent(rechercheIsolee,
                   (ActionEvent.ACTION_PERFORMED, "*")));
        }
    }
);
```

Le code Java (main)

```
public static void main(String[] args) {  
    Frame fenetre = new Frame();  
    DeclencheServlet elle = new DeclencheServlet();  
    fenetre.add(elle);  
    elle.init();  
    elle.start();  
    fenetre.setLocationRelativeTo(null);  
    fenetre.pack();  
    fenetre.setVisible(true);  
    fenetre.addWindowListener(new WindowAdapter() {  
  
        @Override  
        public void windowClosing(WindowEvent arg0) {  
            super.windowClosed(arg0);  
            ((Frame) arg0.getSource()).dispose();  
        }  
    });  
}
```

Enfin, utilisation comme applet



Fichier html :

```
<h1 align="center">Bruits d'animaux</h1>
<p>Cette page web emploie une applet dont le but
    est de permettre la consultation d'une base
    d'informations situ&eacute;e sur un serveur
    distant qui utilise des servlets pour
    r&eacute;agir aux demandes d'informations
    qu'il re&ccedil;oit.</p>
<p>&nbsp; </p>
<applet name="interface"
    code="coursServlets.DeclencheServlet"
    width="500" height="300" />
```

Conclusion

- ❑ Nous avons mis en place une servlet permettant d'effectuer des recherches côté serveur
- ❑ Nous avons mis en place en un seul fichier une application et une applet capables de consulter ce service
- ❑ Les sources complètes sur <http://www.iut-orsay.fr/~fournier>