

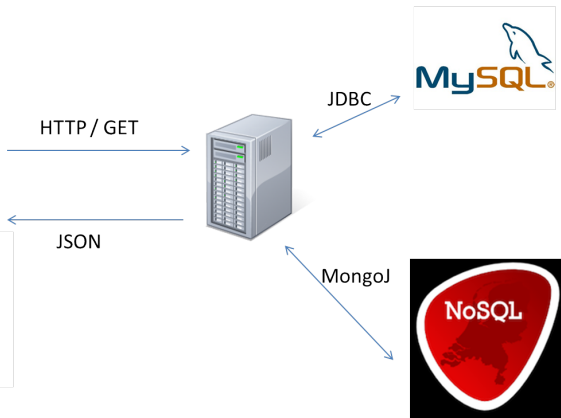
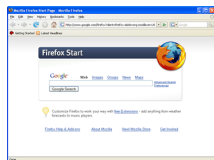
2I017 - Technologies du Web

Ludovic DENOYER - ludovic.denoyer@lip6.fr

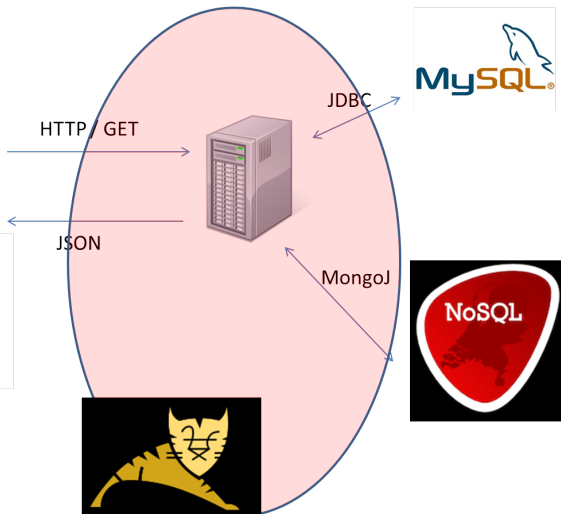
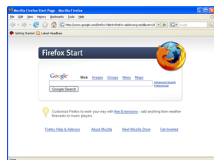
UPMC

20 janvier 2016

Web API



Web API et Tomcat



Apache Tomcat est...

- un conteneur libre de servlets et JSP
- un projet principal de la fondation Apache
- écrit en langage Java, et peut donc s'exécuter via la machine virtuelle Java sur n'importe quel système d'exploitation
- Aujourd'hui : Tomcat 7

Tomcat n'est pas un serveur Web, et est utilisé en parallèle d'un serveur Web classique (Apache).

Servlet

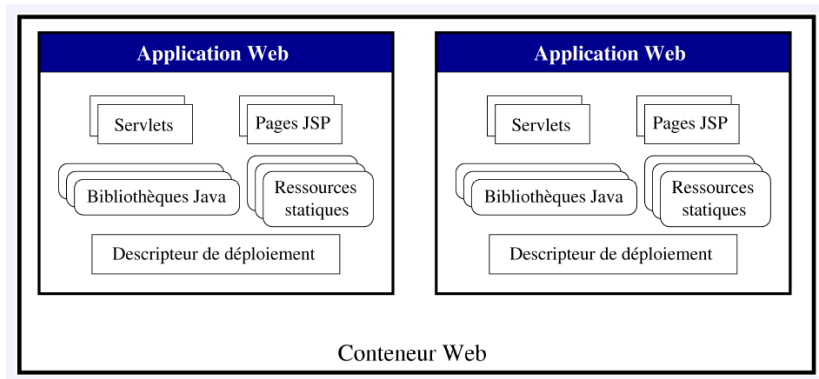
Les Servlets : Définition

- Applications java
- Exécutées côté serveur
- Traitement de requêtes HTTP et génération de réponses dynamiques
 - comparables aux CGI
- Permettent d'étendre les fonctionnalités de base d'un serveur HTTP

Avantages

- Portabilité (java)
- Puissance avec l'accès à la totalité de l'API java
- Rapidité (la Servlet est chargée une seule fois)
- Bons outils de développement (Eclipse)
- Compatible avec vos connaissances :)

Servlet



Servlet : Exemple

Développement de la classe *Hello.java* qui contient la servlet :

HelloWorld.java

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class Hello extends HttpServlet {

    public void doGet(HttpServletRequest request
, HttpServletResponse response)
        throws ServletException,
        IOException
    {
        response.setContentType("text/plain");
        PrintWriter out = response.getWriter();
        out.println("Hello, hello !!!!");
    }
}
```

Servlet : Exemple

Routage de la servlet vers l'URL *bonjour*

web.xml

```
<?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>
<servlet>
<servlet-name>Bonjour</servlet-name>
<servlet-class>Hello</servlet-class>
</servlet>

<servlet-mapping>
<servlet-name>Bonjour</servlet-name>
<url-pattern>/Servlet/Coucou</url-pattern>
</servlet-mapping>
</web-app>
```


Servlet : Exemple

Exportation de l'application Web dans un fichier d'archive de type *WAR* :

Arborescence

```
ApplicationWeb.war
|_ fichiers.html
|_ fichiers.jsp
|_ répertoires/fichiers
|_ META-INF
    |_ MANIFEST.MF
|_ WEB-INF
    |_ web.xml
    |_ classes
    |_ Servlets.class
    |_ lib
        |_ bibliotheques.jar
```

Servlet : Exemple

Déploiement sur le serveur TOMCAT (via l'interface de manager de TOMCAT)

The Apache Software Foundation
http://www.apache.org/

Gestionnaire d'applications WEB Tomcat

Message: OK

Gestionnaire

[Lister les applications](#) [Aide HTML Gestionnaire](#) [Aide Gestionnaire](#) [Etat du serveur](#)

Applications

Chemin	Version	Nom d'affichage	Fonctionnelle	Sessions	Commandes
/	None specified		true	0	Démarrer Arrêter Recharger Retirer Expirer les sessions inactives depuis ≥ 30 minutes
/j328	None specified		true	0	Démarrer Arrêter Recharger Retirer Expirer les sessions inactives depuis ≥ 30 minutes
/TestWeb	None specified		true	0	Démarrer Arrêter Recharger Retirer Expirer les sessions inactives depuis ≥ 30 minutes
/host-manager	None specified	Tomcat Host Manager Application	true	0	Démarrer Arrêter Recharger Retirer Expirer les sessions inactives depuis ≥ 30 minutes
/manager	None specified	Tomcat Manager Application	true	1	Démarrer Arrêter Recharger Retirer Expirer les sessions inactives depuis ≥ 30 minutes

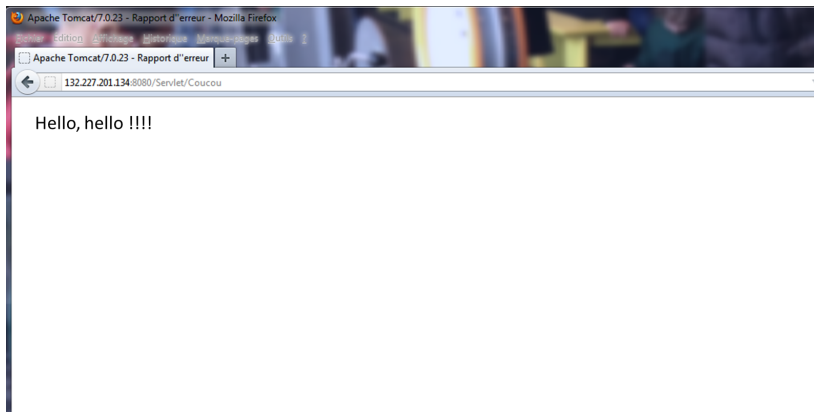
Deployer

Emplacement du répertoire ou fichier WAR de déploiement sur le serveur

Chemin de contexte (requis):

Servlet : Exemple

Appel de la Servlet :



Servlet

Les servlets peuvent "rendre" n'importe quel type de contenu :

- texte
- HTML
- XML
- binaire (fichier, image, son)
- ...
- JSON

La classe HttpServlet

HttpServlet

```
public abstract class HttpServlet extends GenericServlet
{
    HttpServlet()

    // méthodes répondant aux différents types de requêtes
    protected void doDelete(HttpServletRequest req, HttpServletResponse resp)
    protected void doGet(HttpServletRequest req, HttpServletResponse resp)
    protected void doHead(HttpServletRequest req, HttpServletResponse resp)
    protected void doOptions(HttpServletRequest req, HttpServletResponse resp)
    protected void doPost(HttpServletRequest req, HttpServletResponse resp)
    protected void doPut(HttpServletRequest req, HttpServletResponse resp)
    protected void doTrace(HttpServletRequest req, HttpServletResponse resp)

    // permet d'utiliser le cache côté client
    protected long getLastModified(HttpServletRequest req)

    // gère le dispatching en fonction du type de requête}
    protected void service(HttpServletRequest req, HttpServletResponse resp)
    protected void service(ServletRequest req, ServletResponse res)
}
```

Récupération des paramètres

Les paramètres de formulaires sont récupérables via `HttpServletRequest`. La récupération se fait par les méthodes suivantes :

- *Enumeration* `getParameterNames()`
- *String* `getParameter(Stringname)`
- *String[]* `getParameterValues(Stringname)`

Paramètres de Servlet

```
public class HelloPost extends HttpServlet
{
    public void doGet(HttpServletRequest requete, HttpServletResponse reponse)
    throws ServletException, IOException
    {
        String prenom = requete.getParameter("prenom");
        reponse.setContentType("text/html");
        PrintWriter out= reponse.getWriter();
        out.println("<html>");
        out.println("<body>");
        out.println("<h1>Bonjour " + prenom + " ! </h1>");
        out.println("<p>Ceci est ma premiere Servlet...</p>");
        out.println("</body>");
        out.println("</html>");
    }
}
```

URL associée :

<http://.../MyServlet/Hello&prenom=michel>

API REST (Wikipedia)

REST

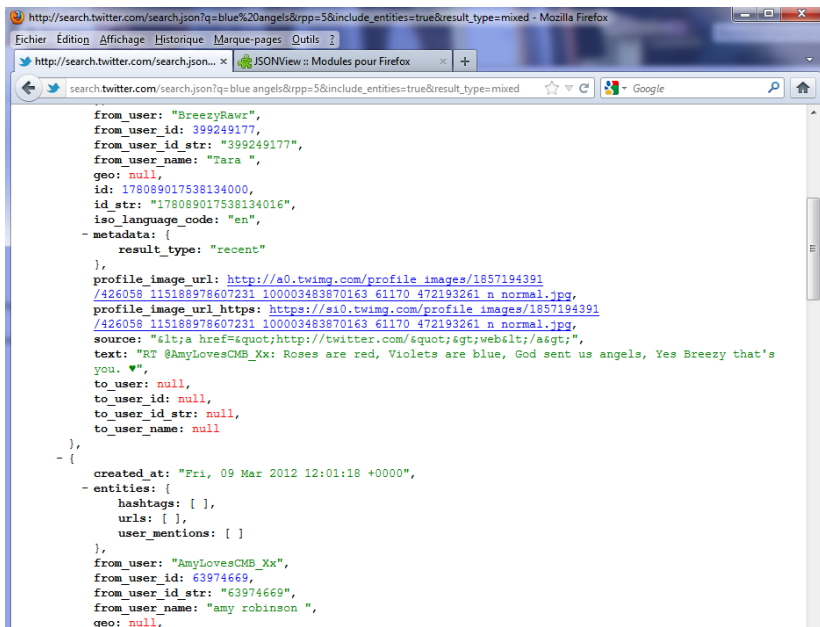
- REST = REpresentational State Transfer
- C'est une manière de construire une application pour les systèmes distribués
- REST n'est pas un protocole ou un format, c'est un style d'architecture
- Il est de plus en plus utilisé pour la réalisation d'architectures orientées services utilisant des services Web destinés à la communication entre machines.

Principes

- l'URI est important : connaître l'URI doit suffire pour nommer et identifier une ressource.
- HTTP fournit toutes les opérations nécessaires (GET, POST, PUT et DELETE, essentiellement).
- Chaque opération est auto-suffisante : il n'y a pas d'état.
- Utilisation des standards hypermedia : HTML ou XML ou **JSON**

Référence : RESTful Web Services, par Leonard Richardson et Sam Ruby

API REST : Exemple Twitter



```
from_user: "BreezyRawr",
from_user_id: 399249177,
from_user_id_str: "399249177",
from_user_name: "Tara ",
geo: null,
id: 178089017538134000,
id_str: "178089017538134016",
iso_language_code: "en",
- metadata: {
  result_type: "recent"
},
profile_image_url: http://a0.twimg.com/profile\_images/1857194391/426058\_115188978607231\_100003483870163\_61170\_472193261\_n\_normal.jpg,
profile_image_url_https: https://s10.twimg.com/profile\_images/1857194391/426058\_115188978607231\_100003483870163\_61170\_472193261\_n\_normal.jpg,
source: "<a href='\"http://twitter.com/\"'>&gt;web</a>",
text: "RT @AmyLovesCMB_Xx: Roses are red, Violets are blue, God sent us angels, Yes Breezy that's you. ♥",
to_user: null,
to_user_id: null,
to_user_id_str: null,
to_user_name: null
},
- {
  created_at: "Fri, 09 Mar 2012 12:01:18 +0000",
  - entities: {
    hashtags: [ ],
    urls: [ ],
    user_mentions: [ ]
  },
  from_user: "AmyLovesCMB_Xx",
  from_user_id: 63974669,
  from_user_id_str: "63974669",
  from_user_name: "amy robinson ",
  geo: null,
```

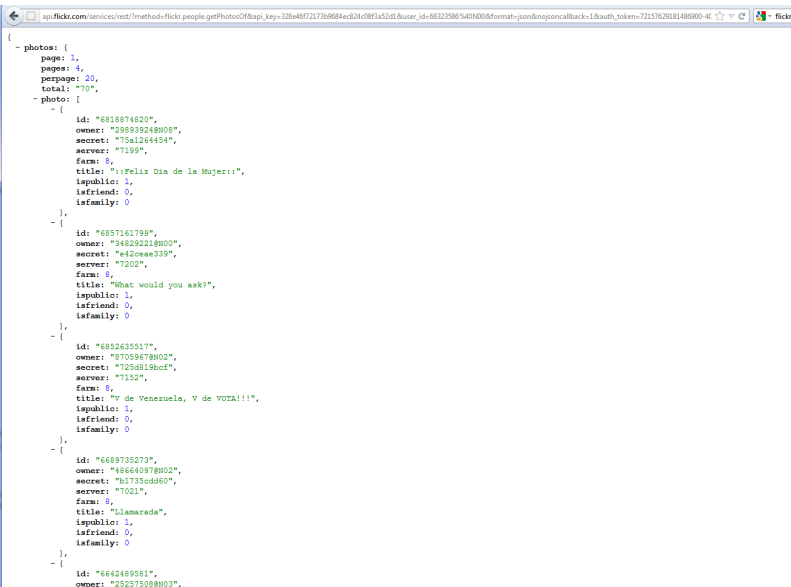
API REST : Exemple Flickr

api.flickr.com/services/rest/?method=flickr.people.getPhotosOf&api_key=328e60f72173b6684ec824c08fa52d1&user_id=68323586%40N00&format=rest&auth_token=72157629181486900-40a224ee68e4346&aj

Aucune information de style ne semble associée à ce fichier XML. L'arbre du document est affiché ci-dessous.

```
<?xml version="1.0"?>
<rsp stat="ok">
  <photos page="1" pages="4" perpage="20" total="70" has_next_page="1">
    <photo id="6818874820" owner="29893924@N08" secret="75a1264454" server="7199" farm="8" title="Feliz Dia de la Mujer." ispublic="1" isfriend="0" isfamily="0"/>
    <photo id="6857161799" owner="34829221@N00" secret="e42ceac339" server="7202" farm="8" title="What would you ask?" ispublic="1" isfriend="0" isfamily="0"/>
    <photo id="6852635517" owner="8705967@N02" secret="725d819bcf" server="7152" farm="8" title="V de Venezuela, V de VOTA!!!" ispublic="1" isfriend="0" isfamily="0"/>
    <photo id="6689735273" owner="48664097@N02" secret="b1735cdd60" server="7021" farm="8" title="Llamarada" ispublic="1" isfriend="0" isfamily="0"/>
    <photo id="6642489581" owner="25257508@N03" secret="a4699ec6d4" server="7025" farm="8" title="Things I Love Thursday" ispublic="1" isfriend="0" isfamily="0"/>
    <photo id="6641187815" owner="48739681@N05" secret="9c5144ca5a" server="7016" farm="8" title="Feliz Nit de Reis - Feliz Noche de Reyes -" ispublic="1" isfriend="0" isfamily="0"/>
    <photo id="6596007895" owner="34829221@N00" secret="961ed2d366" server="7165" farm="8" title="2011 in Review" ispublic="1" isfriend="0" isfamily="0"/>
    <photo id="6593866295" owner="38838924@N03" secret="f867c83f2d" server="7153" farm="8" title="Mi 2011 en flickr" ispublic="1" isfriend="0" isfamily="0"/>
    <photo id="6590116673" owner="8705967@N02" secret="23bd13d66b" server="7018" farm="8" title="Navidad" ispublic="1" isfriend="0" isfamily="0"/>
    <photo id="6587696631" owner="8332135@N03" secret="63cc7b496a" server="7142" farm="8" title="The sky is the limit." ispublic="1" isfriend="0" isfamily="0"/>
    <photo id="6586167143" owner="63307805@N00" secret="cfaeb6d377" server="7007" farm="8" title="my year in pictures" ispublic="1" isfriend="0" isfamily="0"/>
    <photo id="6581955027" owner="39307146@N08" secret="7b56b63043" server="7171" farm="8" title="My year in pictures" ispublic="1" isfriend="0" isfamily="0"/>
    <photo id="6566465059" owner="23523125@N08" secret="4c73a0eb85" server="7147" farm="8" title="Feliz Navidad..." ispublic="1" isfriend="0" isfamily="0"/>
    <photo id="6564959719" owner="29219049@N06" secret="420005be9d" server="7153" farm="8" title="51/52- Feliz Navidad!!!" ispublic="1" isfriend="0" isfamily="0"/>
    <photo id="6560620287" owner="40654531@N05" secret="8f0dc0970a" server="7001" farm="8" title="mi navidad, tu navidad? Explored Dec 23, 2011 #6 Muchas Gracias!" ispublic="1" isfriend="0" isfamily="0"/>
    <photo id="6553379789" owner="49762065@N08" secret="20b62858fe" server="7002" farm="8" title="." ispublic="1" isfriend="0" isfamily="0"/>
    <photo id="6538048271" owner="37763325@N08" secret="66c295ff74" server="7171" farm="8" title="Felices Fiestas! ~ Happy Holidays!" ispublic="1" isfriend="0" isfamily="0"/>
    <photo id="6526003925" owner="66881369@N06" secret="ebb8093294" server="7020" farm="8" title="Family lunch" ispublic="1" isfriend="0" isfamily="0"/>
    <photo id="6475299671" owner="43616712@N03" secret="aa8110b0f7" server="7146" farm="8" title="lovely support" ispublic="1" isfriend="0" isfamily="0"/>
    <photo id="6358869783" owner="25257508@N03" secret="2051f88c38" server="6019" farm="7" title="Friday Favorites" ispublic="1" isfriend="0" isfamily="0"/>
  </photos>
</rsp>
```

API REST : Exemple Flickr



```
{
  - photos: [
    {
      page: 1,
      pages: 4,
      perpage: 20,
      total: "70",
    }
    - photo: [
      - {
        id: "6818874820",
        owner: "296939248N08",
        secret: "75a1264454",
        server: "7199",
        farm: 8,
        title: "¡Feliz Día de la Mujer!",
        ispublic: 1,
        isfriend: 0,
        isfamily: 0
      },
      - {
        id: "6857161799",
        owner: "348292218N00",
        secret: "e42ceae339",
        server: "7202",
        farm: 8,
        title: "What would you ask?",
        ispublic: 1,
        isfriend: 0,
        isfamily: 0
      },
      - {
        id: "6852635517",
        owner: "87059678N02",
        secret: "725d819b0f",
        server: "7152",
        farm: 8,
        title: "V de Venezuela, V de VOTA!!!",
        ispublic: 1,
        isfriend: 0,
        isfamily: 0
      },
      - {
        id: "6689735273",
        owner: "486640978N02",
        secret: "b1735cdd60",
        server: "7021",
        farm: 8,
        title: "Llamarada",
        ispublic: 1,
        isfriend: 0,
        isfamily: 0
      },
      - {
        id: "6642489581",
        owner: "252575088N03",

```

API REST : Exemple Flickr

www.flickr.com/services/api/

- JSON
- PHP

Kits API

Remarque : Les kits API ne sont pas entretenus par Flickr et Flickr n'endosse aucune responsabilité quant à leur utilisation. Pour obtenir de l'aide sur le kit API, rejoignez la [liste de diffusion des développeurs](#) ou contactez directement les personnes chargées de la maintenance.

ActionScript

- api flickr (docs)
- Flashr
- Interfaces REST de l'API Flickr
- as3 flickr lib

C

- Flickcurl

Cold Fusion

- CFlickr

Common Lisp

- Clickr

cUrl

- Curlr

Delphi

- dFlickr

Go

- go-flickr

Java

- flickrj
- flickr-jandroid
- jiclr

.NET

- Flickr.NET

Objective-C

- ObjectiveFlickr

Perl

- Flickr-API2
- Flickr:Upload

PHP

- PEAR: Flickr_API
- phpFlickr

PHP5

- Phlickr

Python

favorites

- flickr.favorites.add
- flickr.favorites.getContext
- flickr.favorites.getList
- flickr.favorites.getPublicList
- flickr.favorites.remove

galleries

- flickr.galleries.addPhoto
- flickr.galleries.create
- flickr.galleries.editMeta
- flickr.galleries.editPhoto
- flickr.galleries.editPhotos
- flickr.galleries.getInfo
- flickr.galleries.getList
- flickr.galleries.getListForPhoto
- flickr.galleries.getPhotos

groups

- flickr.groups.browse
- flickr.groups.getInfo
- flickr.groups.search

groups.members

- flickr.groups.members.getList

groups.pools

- flickr.groups.pools.add
- flickr.groups.pools.getContext
- flickr.groups.pools.getGroups
- flickr.groups.pools.getPhotos
- flickr.groups.pools.remove

interestingness

- flickr.interestingness.getList

machinetags

- flickr.machinetags.getNamespaces
- flickr.machinetags.getPairs
- flickr.machinetags.getPredicates
- flickr.machinetags.getRecentValues
- flickr.machinetags.getValues

API REST : Exemple Flickr

Returns the comments for a photo

Authentication

Cette méthode n'exige pas d'authentification.

Arguments

api_key (Obligatoire)

Your API application key. [See here](#) for more details.

photo_id (Obligatoire)

The id of the photo to fetch comments for.

min_comment_date (Facultatif)

Minimum date that a comment was added. The date should be in the form of a unix timestamp.

max_comment_date (Facultatif)

Maximum date that a comment was added. The date should be in the form of a unix timestamp.

Exemple de réponse

```
<comments photo_id="109722179">
  <comment id="6065-109722179-72057594077818641" author="35468159852@N01" authorname="Rev Dan Catt"
</comments>
```

Codes d'erreur

1: Photo not found

The photo id was either invalid or was for a photo not viewable by the calling user.

100: Invalid API Key

The API key passed was not valid or has expired.

105: Service currently unavailable

The requested service is temporarily unavailable.

111: Format "xxx" not found

The requested response format was not found.

API REST : Principes

Principes Généraux

- Les paramètres sont passés (à travers HTTP) en utilisant le protocole GET (le plus souvent)
- La réponse est un document sous forme JSON ou XML

Avantages

- Simplicité
- Lisibilité par l'humain
- Evolutivité
- Repose sur les principes du Web
- Représentations multiples

Avantages

- Simplicité
- Lisibilité par l'humain
- Evolutivité
- Repose sur les principes du Web
- Représentations multiples

Inconvénients

- Sécurité restreinte par l'emploi de HTTP
- Cible uniquement l'appel de ressources
 - ▶ Architecture orientée ressources (ROA)
 - ▶ ou Architecture orientée données (DOA)

REST vs SOAP

Les Services Web étendus (SOAP) et les Services Web REST sont différents par le fait que :

- Services Web étendus (SOAP)
- Avantages
 - ▶ Standardisé
 - ▶ Interopérabilité
 - ▶ Sécurité (WS-Security)
 - ▶ Outillé
- Inconvénients
 - ▶ Performances (enveloppe SOAP supplémentaire)
 - ▶ Complexité, lourdeur
 - ▶ Cible l'appel de service

TOMCAT et REST

```
@WebServlet("/Example")
public class Example extends HttpServlet
{
    ....

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException
    {
        response.setContentType("text/plain");
        String prenom=request.getParameter("prenom");
        if (prenom==null)
        {
            JSONObject json=new JSONObject();
            try {
                json.put("error","Missing parameter");
            } catch (JSONException e) {
                e.printStackTrace();
            }
            response.getWriter().println(json.toString());
        }
        else
        {
            JSONObject json=new JSONObject();
            try {
                json.put("output","OK");
            } catch (JSONException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
            response.getWriter().println(json.toString());
        }
    }
}
```

Conclusion

Conclusion

Vous connaissez :

- TOMCAT
- REST

TD / TP

- Spécifier les services à développer
- Mettre en place l'architecture des services (en attendant l'accès au BDs)

Prochain Cours : SQL et JDBC