

ENSM-SE

Axe Ingénierie des Systèmes Informatiques  
Ecole Nationale Supérieure des Mines de Saint-Etienne

Développements Web en Java  
Développement de Systèmes Informatiques

© 2009, Laurent Vercoeur

Ecole Nationale Supérieure des Mines  
SAINT-ETIENNE

---

---

---

---

---

---

---

---

ENSM-SE

plan du cours

- Lundi : cours XML
- Mardi matin : TP XML
- Mardi après-midi : Introduction aux serveurs web (Apache Tomcat), programmation de servlet
- Mercredi : JavaBeans, Java Server Pages
- Jeudi : interaction avec une BDD (JDBC)
- Vendredi matin : contrôle de versions (SVN)
- Vendredi après-midi : TP d'intégration de ces frameworks

© 2009, Laurent Vercoeur

Ecole Nationale Supérieure des Mines  
SAINT-ETIENNE

---

---

---

---

---

---

---

---

ENSM-SE

introduction au développement web

- Frameworks pour le déploiement d'application Java sur le web
  - ◆ Bénéfice des protocoles de communication du web
  - ◆ Réseau large échelle (internet, intranet, extranet)
  - ◆ Ouvre toutes les possibilités d'un langage de programmation généraliste
- Types de frameworks existants
  - ◆ Déploiement d'objets Java (Servlets, JSP)
  - ◆ Pattern modèle-Vue-Contrôleur (Struts, Tapestry, JSF)
  - ◆ Intégration de technologies XML (Ajax)
  - ◆ Connexion avec des BDDs (Hibernate, JPA, JDO)

© 2009, Laurent Vercoeur

Ecole Nationale Supérieure des Mines  
SAINT-ETIENNE

---

---

---

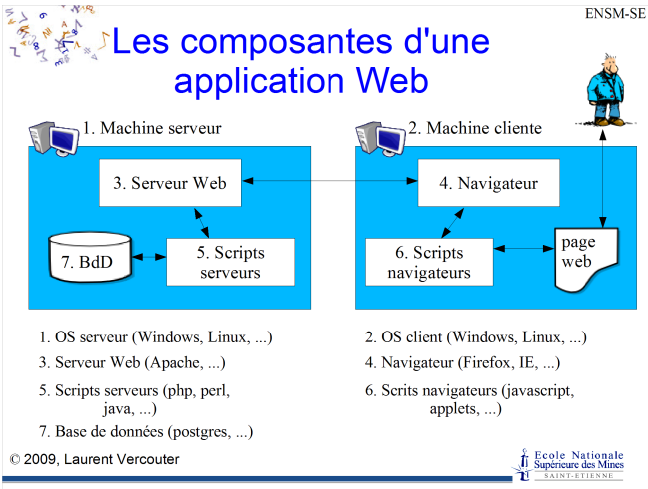
---

---

---

---

---



## Pages HTML statiques/dynamiques

Une page web est

- **statique** si toutes les informations nécessaires à sa création sont indiquées à l'écriture du fichier
  - ◆ exemple : fichier HTML
- **dynamique** si elle est générée « à la volée »
  - ◆ exemples : fichiers ASP, Perl, php

© 2009, Laurent Vercouter

## Exemple de page statique

maPage.html:

```
<html>
  <head>
    <title>une page statique</title>
  </head>
  <body>
    <center>
      <h1>Une page statique...</h1>
    </center>
  </body>
</html>
```

© 2009, Laurent Vercouter

---

---

---

---

---

---

---

---

---

---



---

---

---

---

---

---

---

---

---

---



---

---

---

---

---

---

---

---

---

---

## Exemple de page dynamique

maPage.php :

```
<html>
<head>
  <title>une page dynamique</title>
</head>
<body>
  <h1>Une page PHP générée dynamiquement</h1>
  <h2>
    <?
      $maintenant=time();
      echo date("j/m/y, h:i:s", $maintenant);
    ?>
  </h2>
</body>
</html>
```

## Applets Java

- Applet = logiciel capable de s'exécuter dans une page HTML
  - ◆ S'exécute sur la machine du client
  - ◆ Accès restreints (système de fichier)
- Développement d'une applet
  - ◆ Classe héritée de javax.swing.JApplet
  - ◆ Introduite dans la page HTML par le tag <applet>
 

```
<APPLET CODE=AppletSubclass.class WIDTH=anInt
          HEIGHT=anInt>
          <PARAM NAME=parameter1Name VALUE=aValue>
          <PARAM NAME=parameter2Name VALUE=anotherValue>
        </APPLET>
```

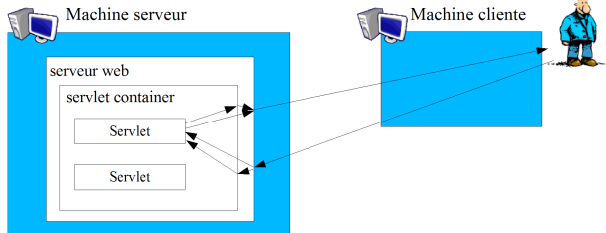
## Servlets et JSP

La génération d'une page dynamique peut se faire en Java à l'aide de servlets ou de JSP

- **Servlet**
  - ◆ Un servlet container inclus dans le serveur web oriente les requêtes réseau vers les servlets
  - ◆ Un servlet est déployé dans un servlet container et répond aux requêtes en générant dynamiquement une page
- **Java Server Pages (JSP)**
  - ◆ Langage de script qui réduit la portion de code Java à écrire
  - ◆ Génère automatiquement des servlets

# Servlet Container

- Un Servlet container contient des servlets et dirige les requêtes et les réponses

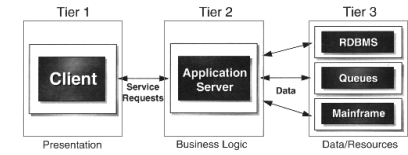


## Serveur web Apache Tomcat

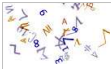
# Architecture 3-tiers

Client allégé – serveur de traitement – serveur de données

- Découplage des parties
  - ♦ Clientes
  - ♦ Logique Métier
  - ♦ Données Métier







## Solutions techniques

Frameworks pour gérer les différentes couches :

- Couche **Présentation** : Struts/WebWork, JSF, Tapestry, GWT, ...
- Couche **Métier** : Spring/Hivemind, Expresso, EJB Session, Business Rules, ...
- Couche **Persistance** : EJB, Hibernate, iBatis, JDO, Oracle, ...
- Communications : HTTP, SOAP, JMS, RMI, ...

---

---

---

---

---

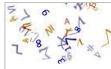
---

---

---

---

---



## Apache Tomcat

- Tomcat est un serveur d'application et un serveur Web
  - ♦ Serveur web : traite les requêtes HTTP
  - ♦ Serveur d'applications : gère des applications
- Tomcat 6.x téléchargeable : <http://tomcat.apache.org/>
- Lancement de Tomcat : `bin/startup`
- Arrêt de Tomcat : `bin/shutdown`
- Le serveur web lancé peut être vu par un navigateur à l'URL : `http://localhost:8080`

---

---

---

---

---

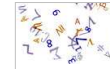
---

---

---

---

---



## Tomcat : Concepts de base

- Tomcat regroupe un ensemble de contextes (ex: contexte ISI)
  - ♦ Correspond au répertoire `$TOMCAT_HOME$/webapps/ISI`
  - ♦ Correspond à l'URL <http://localhost:8080/ISI>
- Un contexte contient
  - ♦ Des pages statiques
  - ♦ Des JSP
  - ♦ Des servlets
  - ♦ Un répertoire WEB-INF contenant fichiers de configuration, ressources, classes Java

---

---

---

---

---

---

---

---

---

---

## Configuration d'un contexte

- Fichier WEB-INF/web.xml

- ♦ Décrit le contexte
- ♦ Répertorie les servlets

```
<servlet>
  <servlet-name>HelloWorldExample</servlet-name>
  <servlet-class>HelloWorldExample</servlet-class>
</servlet>

<servlet-mapping>
  <servlet-name>HelloWorldExample</servlet-name>
  <url-pattern>/applis/HelloWorldExample</url-pattern>
</servlet-mapping>
```

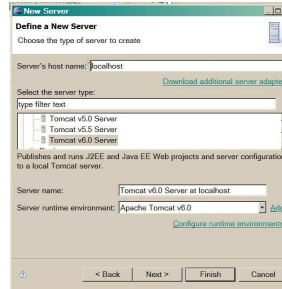
- ♦ Définit des mappings d'accès aux servlets

- Répertoire WEB-INF/classes

- ♦ Contient les classe Java

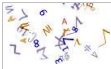
## Tomcat sous Eclipse

- Déclaration d'un serveur : New... -> Server -> Server



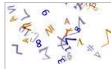
- Création d'un contexte : New... -> Dynamic Web project

## Servlets



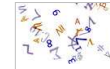
## API javax.servlet

- les packages de `javax.servlet` sont inclus dans la version actuelle de J2EE ainsi que dans les distributions de Tomcat (rep. lib)
- interface `javax.servlet.Servlet`
  - `init(ServletConfig)`
  - `destroy()`
  - `service(ServletRequest, ServletResponse)`
  - `ServletConfig getConfig()`
  - `String getServletInfo()`



## Méthode init()

- Appelée au chargement du servlet
  - au lancement du serveur
  - si le servlet a été modifié
- Le paramètre d'entrée `ServletConfig` encapsule les informations nécessaires à l'initialisation



## Méthode destroy()

La méthode `destroy()` est appelée lorsque le servlet est "déchargé" du serveur web.

Généralement, son traitement consiste à libérer les ressources et fermer les connexions créées à l'initialisation du servlet.

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

## Méthode service()

- Appelée à chaque fois qu'une requête est transmise au servlet
  - ♦ Le paramètre ServletRequest encapsule les informations relatives à la requête
    - Enumeration `getParameterNames()`
    - `String[]` `getParameterValues(String)`
    - `String` `getParameter(String)`
  - ♦ Le paramètre ServletResponse sert à remplir la réponse du servlet
    - `void` `setContentType(String)`
    - `OutputStream` `getOutputStream()`

## javax.servlet.ServletRequest

- Les méthodes de l'interface ServletRequest permettent d'accéder à des informations sur une requête.
  - ♦ `String[]` `getParameterNames()`
    - renvoie l'ensemble des noms de paramètres
  - ♦ `String[]` `getParameterValues(String)`
    - renvoie toutes les valeurs d'un paramètre
  - ♦ `String` `getParameter(String)`
    - renvoie la valeur d'un paramètre

## javax.servlet.ServletResponse

- Les méthodes de l'interface ServletResponse permettent d'envoyer des informations à un client ayant émis une requête.
  - ♦ `setContentType(String)`
    - définit le type de contenu
    - EX: `res.setContentType("text/html");`
  - ♦ `OutputStream` `getOutputStream()`
    - renvoie le flux de sortie vers le client
    - EX: `PrintWriter out = new PrintWriter(res.getOutputStream());`  
`out.println("<html><body>hello</body></html>");`

ENSM-SE

## Exemple « HelloWorld »


```
import javax.servlet.*;

public class HelloWorld implements Servlet {
    ...
    public void init(ServletConfig c) { ... }

    public void destroy() { }

    public void service(ServletRequest req, ServletResponse res)
        throws ServletException, IOException {
        res.setContentType("text/html");
        PrintWriter out = res.getWriter();
        out.println("<html>\n<body>\n<h1>Hello world</h1>");
        out.println("</body>\n</html>");
        out.close();
    }
}
```

© 2009, Laurent Vercouter




ENSM-SE

## protocole HTTP & Servlets

- La plupart des servlets sont conçues pour répondre à des requêtes HTTP
- la classe `javax.servlet.http.HttpServlet` implémente l'interface `javax.servlet.Servlet` pour le protocole HTTP
  - ◆ GET
  - ◆ HEAD
  - ◆ POST
  - ◆ PUT
  - ◆ DELETE
  - ◆ ...

© 2009, Laurent Vercouter




ENSM-SE

## `javax.servlet.http.HttpServlet`

L'implémentation de la méthode service détermine le type de requête et invoque la méthode correspondante

- ◆ `doGet(HttpServletRequest, HttpServletResponse)`
- ◆ `doPost(HttpServletRequest, HttpServletResponse)`
- ◆ `doHead(HttpServletRequest, HttpServletResponse)`
- ◆ `doPut(HttpServletRequest, HttpServletResponse)`
- ◆ `doOptions(HttpServletRequest, HttpServletResponse)`

© 2009, Laurent Vercouter



---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

## Ecriture d'un objet HttpServlet

- Héritage de la classe HttpServlet
  - surcharge de quelques méthodes do...()
  - ne surtout pas surcharger la méthode service() !

```
protected void doGet(HttpServletRequest req,
    HttpServletResponse resp) throws
    ServletException, IOException {
    resp.setContentType("text/html");
    PrintWriter out = resp.getWriter();
    out.println("<html>\n<body>\n<h1>Hello World</h1>");
    out.println("</body>\n</html>");
    out.close();
}
```

## javax.servlet.HttpServletRequest

- Les méthodes de l'interface HttpServletRequest permettent d'accéder à des informations sur une requête HTTP.
  - String getRemoteUser()
    - renvoie le nom de l'utilisateur de la requête
  - String getMethod()
    - renvoie le type (get, post, ...) de la requête HTTP
  - ainsi que les méthodes de ServletRequest

## javax.servlet.HttpServletResponse

- Les méthodes de l'interface HttpServletResponse permettent d'envoyer des informations à un client ayant émis une requête HTTP.
  - sendError(int)
    - renvoie une erreur
  - sendRedirect(String)
    - renvoie une redirection vers une autre adresse
  - ainsi que les méthodes de ServletResponse