

# Web Server et Servlets.

*Nous abordons ici la partie 'serveur' des applications distribuées au travers des spécifications JEE.*

## 1 Introduction

---

Un Http Serveur (ou Web Server) est un logiciel fournissant à ses clients au travers d'internet (internet, intranet, extranet) et de http un ensemble de ressources.

Ces ressources peuvent être des pages Html, des images, des documents quelconques (généralement XML), ... Ces différentes ressources peuvent être préexistantes [on parle alors de ressources statiques] ou construites à la demande [il s'agira alors de ressources dynamiques]. Le Web serveur peut faire appel à d'autres serveurs pour obtenir le contenu des ressources à construire.

Un des rôles du Web Serveur sera de faire la correspondance entre l'url fournie par le client et la ressource elle-même (ou son mécanisme de construction). Un tel serveur est généralement doté de mécanismes d'authentification et de cryptage pour assurer la sécurité.

Nous allons découvrir cette notion au travers des spécifications JEE [ou Java Enterprise Edition]. D'autres environnements existent (.Net de microsoft par exemple). Cette année nous utiliserons Tomcat de la fondation Apache qui intègre un Web server et implémente les spécifications JEE des servlets et des jsp.

Dans un premier temps nous allons voir comment mettre en œuvre ce serveur pour fournir des pages statiques.

Dans un deuxième temps, nous verrons comment nous pouvons lui faire réaliser des traitements particuliers comme de construire des pages HTML dont le contenu sera fourni par d'autres serveurs ou applications.

## 2 Premiers exemples

---

Sous netbeans, créons un projet (de type « application web ») et appelons le « premierWeb » sans utiliser encore de Framework (nous aborderons cette notion l'an prochain).

### 2.1 Définissons nos premières ressources statiques

a) Créons une page HTML 'bonjour à tous' que nous appellerons Welcome.html. Plaçons ce fichier dans web pages. Editons le fichier de configuration 'web.xml' pour spécifier que cette page sera la page initiale [Onglet 'Pages'].

Lançons le serveur en n'oubliant pas de cocher la case 'http monitor', faites le build de l'application et lancez le 'run' qui va construire l'application, la déployer sur le serveur et ouvrir votre browser en lui fournissant l'url de l'application.

Nous voyons que l'adresse de l'application est <http://localhost:8080/premierWeb/> correspondant au

context-path visible dans les propriétés de l'application [Run].

Vérifier que vous pouvez aussi accéder à la page composée par votre voisin en modifiant l'url dans votre browser.

Ajoutons un css pour notre page. Prenons déjà l'habitude d'organiser notre arborescence et plaçons ce css dans un répertoire spécifique aux css. Pour ajouter le css à votre application, soit vous créez un nouveau document et vous y collez le contenu du css soit vous faites un transfert de fichier vers le répertoire où il doit être placé.

Faisons une référence à une image (logo de l'Esi par exemple) dans le css. Plaçons cette image dans un répertoire spécialisé. Testons.

b) Rédigeons une deuxième page (Apropos.html) de présentation de l'application et de l'auteur que nous plaçons dans un répertoire « Apropos » du répertoire « Web Pages ». Appelons cette page à partir de notre page d'accueil au travers de 2 balises `<a href='...'>llk</a>` (l'une avec une référence relative, l'autre avec une référence absolue).

**Remarque :** dans l'onglet http monitor de Netbeans, nous pouvons aller voir l'ensemble des requêtes http que le serveur a traitées. Ce petit moniteur nous sera souvent utile pour comprendre d'éventuels dysfonctionnements de notre développement.

### 3 Quelques éléments de l'environnement

---

Java Enterprise Edition, JEE (anciennement appelé J2EE) est un ensemble de spécifications liées à Java pour le développement d'applications distribuées. Nous nous contenterons d'aborder cette année quelques notions liées à la partie Web de JEE: les notions de servlets et de JSP.

#### 3.1 L'arborescence de Web pages

L'arborescence de Web pages que nous serons amenés à développer est fondamentale pour organiser proprement notre application et mettre en œuvre la sécurité. WEB-INF y joue un rôle essentiel: toutes les ressources qui y sont placées ne pourront jamais être atteintes directement par le client mais le web server pourra y accéder pour nourrir les documents qu'il composera pour le client. C'est au travers du fichier de description web.xml que nous pourrons, entre autres spécifications, 'mapper' [faire correspondre] les url fournies par le client et les ressources dont dispose l'application. Ce fichier peut être édité et modifié soit directement soit au travers d'une interface fournie par NetBeans. Les caractéristiques de ce fichier sont définies par JEE.

#### 3.2 La sécurité

Java EE fournit des outils de sécurisation de nos applications.

Nous avons la possibilité d'authentifier un utilisateur qui s'identifie et de contrôler – de manière déclarative – ses accès aux ressources.

L'authentification peut être réalisée par le serveur JEE (qui intègre un serveur d'authentification) ou déléguée à un serveur d'authentification tiers (windows, ldap, ...).

Les autorisations octroyées à des utilisateurs ou à des groupes d'utilisateurs ressources se donneront sur les répertoires de Web pages.

Nous verrons plus tard comment nous pouvons mettre en œuvre cette sécurité mais il est important de percevoir que la manière de répartir les ressources dans l'arborescence est significative.

### 3.3 Les servlets

Les éléments abordés jusqu'ici concernaient des ressources statiques.

Une servlet, notion JEE, est une classe java qui a pour environnement d'exécution un conteneur de servlets et a pour but de participer à la génération de ressources dynamiques. Les différentes servlets d'un conteneur constituent une Web Application.

### 3.4 Les JSP

JavaServer Pages est une technologie permettant de générer des servlets à partir de documents (pages jsp) intégrant de manière lisible du code HTML et du code java.

## 4 Exemples de servlets

---

Attention cet exemple n'a un sens que dans ce cheminement de découverte de la notion de servlet. En pratique, personne n'écrira une servlet de ce type.

### 4.1 Hello World

Au niveau de l'application, faisons New – Servlet. Appelons notre Servlet `PremiereServlet` et plaçons-la dans un package `gxxx.mesServlets`. NetBeans nous demande si nous désirons déjà réaliser le mapping de la ressource avec une url. Gardons sa proposition par défaut qui sera automatiquement ajoutée au fichier `web.xml` [allons le voir]. N'oublions pas que ce mapping devra être fait en tenant compte des impératifs de sécurité.

Retirons les commentaires et modifions le code comme ceci :

```
public class PremiereServlet extends HttpServlet {

    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Servlet PremiereServlet</title>");
        out.println("<link rel='stylesheet' type='text/css' href='CSS/Welcome.css' />");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>Servlet PremiereServlet at " + request.getContextPath () + "</h1>");
        out.println("<br/><br/><p>Nous sommes le "+new Date()+"</p>");
        out.println("</body>");
        out.println("</html>");
        out.close();

        ...
    }
}
```

Faisons le run de l'application et modifions l'url demandée pour accéder à la page composée par le serveur. Remarquons que la date affichée est la date du serveur [dans notre cas ceci n'a guère de sens puisque notre serveur et notre client partagent le même hôte].

## 4.2 Exploitation de données saisies

Ecrivons un formulaire html fournissant nos nom et prénom et écrivons une servlet générant une page html présentant les valeurs reçues ainsi que des informations liées à notre poste client (ip address, browser, OS, ...).

```
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    PrintWriter out = response.getWriter();
    out.println("<html>");
    out.println("<head>");
    out.println("<title>Servlet BienRecu</title>");
    out.println("</head>");
    out.println("<body>");
    out.println("<h1>Servlet BienRecu at " + request.getContextPath () + "</h1>");
    out.print("nom: "+request.getParameter("nom")+"<br/>");
    out.print("prénom: "+request.getParameter("prenom")+"<br/>");
    out.print("<br/><br/>");
    out.print("Mon adresse ip est: "+request.getRemoteAddr()+"<br/>");
    out.print("Info Browser: "+request.getHeader("user-agent")+"<br/>");
    out.println("</body>");
    out.println("</html>");

    out.close();
}
```

Exercice : complétez votre formulaire pour faire en sorte que nom et prénom soient obligatoires.

## 4.3 Appel à une application extérieure

Ajoutons eVente.jar aux librairies de l'application.

Ecrivons une page rechProduit.html permettant de saisir comme critère de recherche une fourchette de prix et retournant la liste des produits dont le prix effectif est compris dans la fourchette.

```
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    PrintWriter out = response.getWriter();
    try {
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Servlet FouchetteServlet</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>Servlet FouchetteServlet at " + request.getContextPath() + "</h1>");

        String p1 = request.getParameter("prixBas");
        String p2 = request.getParameter("prixHaut");
        ProduitSel sel = new ProduitSel(null, Integer.parseInt(p1), Integer.parseInt(p2), 0, 0, 0, null);
        Collection<ProduitDto> col = VisiteurFacade.getPrdSelectionnes(sel);
        if (col.size() == 0) {
            out.println("<p>Aucun produit ne correspond à votre sélection</p>");
        } else {
            out.println("<table>");
            out.println("<tr>");
            out.println("<th>Marque</th>");
            out.println("<th>Libelle</th>");
            out.println("<th>Prix</th>");
            out.println("</tr>");
        }
    }
}
```

```

        for (ProduitDto prd : col) {
            out.println("<tr>");
            out.println("<td>" + prd.getMarque().getLibelle() + "</td>");
            out.println("<td>" + prd.getLibelle() + "</td>");
            out.println("<td>" + prd.getPrixEffectif() + "</td>");
            out.println("</tr>");
        }
        out.println("</table>");
    }

    } catch (EventeBusinessException ex) {
        out.println("<p class='error' >Une erreur technique nous empêche de vous délivrer le
résultat:<br/>" + ex.getMessage() + "</p>");
    } catch (NumberFormatException nBe){
        out.println("<p class='error' >Les données saisies doivent être numériques!</p>");
    }finally {
        out.println("</body>");
        out.println("</html>");
        out.close();
    }
}

```

Exercice: veuillez à valider la saisie: valeurs obligatoires et numériques. Soignez l'affichage du résultat: présentez les images, faites apparaître clairement les produits en promotion, ...