

Cours Développement d'applications Web et Multimédia

# Chap2 : Composant Web Java EE : Servlets

Filière : 3<sup>ème</sup> année Licence Multimédia

Amen Ajroud

`amen.ajroud.isitcom@gmail.com`

Année Universitaire : 2021-2022

# Plan

- Définition
- Exécution
- Méthodes d'envoi GET et POST
- Implémentation
- Objets request et response
- Annotations

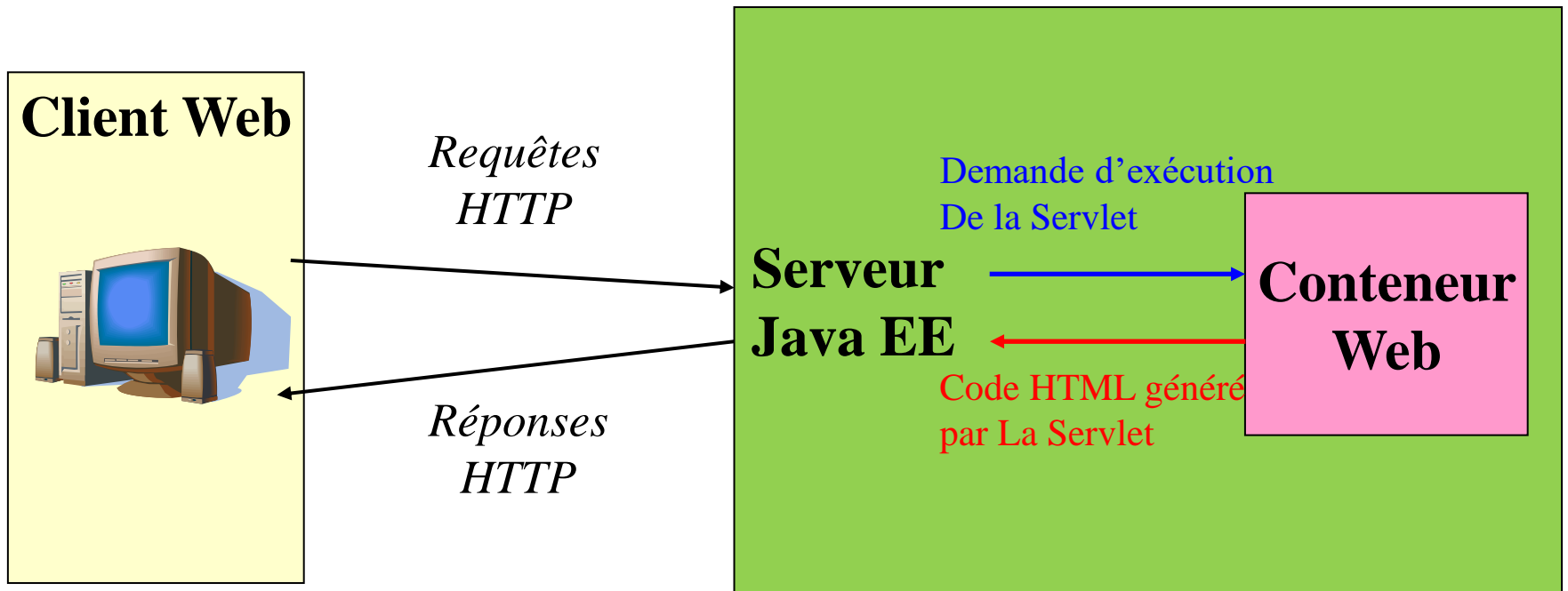
# Définition

- Une Servlet est un composant web Java EE
- C'est une classe Java qui s'exécute dans le conteneur web du serveur Java EE.
- Une Servlet peut:
  - Prendre en charge (accepter) des requêtes HTTP envoyées par des clients web (navigateur),
  - Générer une réponse HTTP pour ce client (en format HTML).

# Définition

- Lorsqu'une URL de Servlet est demandée par un client web (navigateur), le serveur Java EE passe la main au conteneur web pour exécuter le code de la Servlet.
  - L'exécution du code Java de la Servlet génère un contenu HTML.
  - Ce contenu HTML est retourné comme réponse HTTP au client Web.

# Exécution



# Exécution

Il y a 3 façons pour qu'un client web fait appel à une servlet :

- À travers son URL, ex :
  - `http://localhost:8080/mon_app/nom_Servlet`
- A travers un lien hypertexte :
  - `<a href="nom_servlet">texte lien</a>`
- Dans l'attribut action d'un formulaire, ex:
  - `<form method="..." action="nom_Servlet">`

# Méthodes d'envoi d'une requête HTTP

- Méthode `GET` est utilisée par défaut pour l'envoi de toute requête HTTP :
  - les paramètres de la requête sont ajoutées à l'URL .
- Méthode `POST` remplit les mêmes services de `GET` mais
  - les paramètres de la requête sont transmis dans le corps de la requête (masqués).

# Implémentation

- Les servlets (invoquées à partir du protocole HTTP) **héritent** de la classe : `javax.servlet.http.HttpServlet`
- Une servlet doit implémenter l'une des méthodes `doXXX()` pour traiter la requête HTTP reçue :

- Si la méthode d'envoi de la requête HTTP est GET, on définit la méthode :

```
public      void      doGet (HttpServletRequest      request,  
    HttpServletRequest response)
```

- Si la méthode d'envoi de la requête HTTP est POST, on définit la méthode :

```
public      void      doPost (HttpServletRequest      request,  
    HttpServletRequest response)
```



# L'objet requête `HttpServletRequest`

- Les méthodes `doPost (...)` et `doGet (...)` prennent en paramètre un objet de type `HttpServletRequest`
- L'objet de type `HttpServletRequest` contient les informations sur la requête du client et sur l'environnement du serveur.
- Quelques méthodes de cet objet :
  - `String getParameter(name)` : retourne la valeur du paramètre `name`
  - `String[] getParameterValues(name)` : retourne les valeurs du para `name`
  - `String getServletPath()` : retourne l'URL d'appel de la Servlet
  - `Void setAttribute(att, obj)` : ajoute un attribut à la requête
  - `Objet getAttribute(att)` : récupère la valeur de l'attribut de la requête
  - `RequestDispatcher getRequestDispatcher(url)` : redirection interne vers la ressource `url`.

# L'objet réponse `HttpServletResponse`

- L'objet de type `HttpServletResponse` est utilisé pour construire un message de réponse HTTP envoyé au client, il contient :
  - Les méthodes nécessaires pour définir le type du contenu, le code de retour
  - Un flot de sortie pour envoyer les données au client
- Exemple de méthodes :
  - `void setContentType(String)` : définit le type du contenu MIME (ex: `text/html`)
  - `PrintWriter getWriter()` : flot pour envoyer les données au client
  - `void sendRedirect(String)` : redirige le navigateur vers un l'URL en paramètre

# Premier exemple de servlet

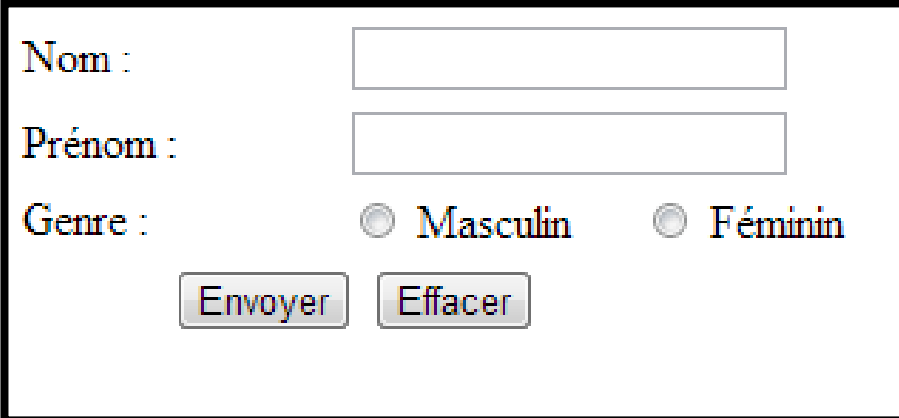
```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class MaPremiereServlet extends HttpServlet {
    public void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<html>");
        out.println("<head><title>Bonjour</title></head>");
        out.println("<body>");
        out.println("<h1> Bonjour à tous </h1>");
        out.println("</body></html>");
    }

    public void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        doGet(request, response);
    }
}
```

# Servlet HTTP et formulaire : Exercice

- Créer un formulaire web qui saisie un nom, prénom et genre (bouton radio)
- Les données seront envoyées via la méthode POST à la servlet `FormServlet`.



The image shows a web form with the following elements:

- A label "Nom :" followed by a text input field.
- A label "Prénom :" followed by a text input field.
- A label "Genre :" followed by two radio buttons. The first radio button is labeled "Masculin" and the second is labeled "Féminin".
- Below the radio buttons are two buttons: "Envoyer" and "Effacer".

# Servlet HTTP et formulaire : Exercice

- Créer le code de la servlet `FormServlet` qui affiche les données saisies dans le formulaire

**Nom : Ben fouleni**  
**Prénom : Foulen**  
**Vous êtes un Homme**

# Servlet HTTP et formulaire : Exercice

```
<form method="post" action="FormServlet">  
  Nom : <input name="nom" type="text" /><br/>  
  Prénom : <input name="prenom" type="text" /><br/>  
  Genre :  
  <input type="radio" name="genre" value="M"/> Masculin  
  <input type="radio" name="genre" value="F"/> Féminin  
  <br/>  
  <input type="submit" name="Submit" value="Envoyer" />  
  <input type="reset" value="Effacer" />  
</form>
```

# Servlet HTTP et formulaire : Exercice

```
public class FormServlet extends HttpServlet{
    public void doPost(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException{
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<html><head></head><body>");
        out.println("<p>Nom : " + request.getParameter("nom"));
        out.println("<p>Prénom : " + request.getParameter("prenom"));
        if(request.getParameter("genre").equals("M"))
            out.println("<p>Vous êtes un Homme</p> ");
        else
            out.println("<p>Vous êtes une Femme</p> ");
        out.println("</body></html>");
    }
}
```

# Annotations

- Une annotation est une méta-donnée, une indication liée à l'élément qu'elle cible.
- Il existe des annotations associées aux Servlets parmi eux l'annotation `@WebServlet`
- L'annotation `@WebServlet` est utilisé pour définir un URL d'appel différent du nom de classe de la Servlet.
- On l'utilise pour masquer à l'utilisateur le nom réel de la classe Servlet (raison de sécurité)



# Annotation @WebServlet : Exemples

- Un seul URL d'appel

```
@WebServlet("/processForm")
```

```
public class MyServlet extends HttpServlet {
```

```
...
```

- Servlet annoté avec deux URLs

```
@WebServlet(urlPatterns = {"/sendFile", "/uploadFile"})
```

```
public class UploadServlet extends HttpServlet {
```

```
...
```