JEE : paramètres d'initialisation de servlet et Filtres

Achref El Mouelhi

Docteur de l'université d'Aix-Marseille
Chercheur en programmation par contrainte (IA)
Ingénieur en génie logiciel

elmouelhi.achref@gmail.com

Plan

ServletContext et ServletConfig

Les paramètres de contexte

- Ce sont des paramètres partagés par toutes les servlets
- Chaque paramètre a un nom et une valeur et peut avoir une description

ServletContext et ServletConfig

Les paramètres de contexte

- Ce sont des paramètres partagés par toutes les servlets
- Chaque paramètre a un nom et une valeur et peut avoir une description

Les paramètres de configuration

- Ce sont des paramètres relatifs à une servlet
- Chaque paramètre a aussi un nom et une valeur et peut avoir une description

ServletContext et ServletConfig

Pour mieux comprendre

- Créer un projet JEE
- Créer deux servlets `FirstServlet` et `SecondServlet`
- Mettre en commentaire l'annotation `@WebServlet` dans les deux fichiers

ServletContext et ServletConfig

Le contenu de `web.xml`

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp
   .org/xml/ns/javaee/web-app_3_1.xsd" id="WebApp_ID" version="3.1">
<servlet>
  <servlet-name>FirstServlet</servlet-name>
  <servlet-class>org.eclipse.controller.FirstServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>FirstServlet</servlet-name>
  <url-pattern>/FirstServlet</url-pattern>
</servlet-mapping>
<servlet>
  <servlet-name>SecondServlet</servlet-name>
  <servlet-class>org.eclipse.controller.SecondServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>SecondServlet</servlet-name>
  <url-pattern>/SecondServlet</url-pattern>
</servlet-mapping>
</web-app>
```

ServletContext et ServletConfig

Pour déclarer des paramètres partagés par toutes les servlets

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp
   .org/xml/ns/javaee/web-app_3_1.xsd" id="WebApp_ID" version="3.1">
<!-- declaration de toutes les servlets -->
<context-param>
  <param-name>nom</param-name>
  <param-value>wick</param-value>
</context-param>
</web-app>
```

ServletContext et ServletConfig

Pour déclarer un paramètre propre à une servlet

```
<!-- le contenu precedent -->
<servlet>
  <servlet-name>FirstServlet</servlet-name>
  <servlet-class>org.eclipse.controller.FirstServlet</servlet-class>
  <init-param>
    <param-name>nom</param-name>
    <param-value>travolta</param-value>
  </init-param>
</servlet>
<servlet-mapping>
  <servlet-name>FirstServlet</servlet-name>
  <url-pattern>/FirstServlet</url-pattern>
</servlet-mapping>
<!-- + la declaration de la seconde servlet -->
```

ServletContext et ServletConfig

Pour tester : contenu de la méthode `doGet()` de `FirstServlet` et `SecondServlet`

```
protected void doGet(HttpServletRequest request,
   HttpServletResponse response) throws ServletException,
    IOException {
  // TODO Auto-generated method stub
  ServletContext  context = getServletContext();
  String nom = context.getInitParameter("nom");
  PrintWriter out = response.getWriter();
  out.print(nom + " from ServletContext");
  nom = getServletConfig().getInitParameter("nom");
  out.println(nom + " from ServletConfig");
}
```

ServletContext et ServletConfig

Remarques

- Il est possible aussi d'utiliser l'annotation `@WebInitParam(name="nom", value="travolta")` pour initialiser les paramètres d'une servlet
- Pour les paramètres du contexte, il n'existe pas d'annotation pour remplacer le `web.xml`

Filters

Les filtres

- un composant d'une application web
- une classe Java très similaire à une servlet, il faut
  - soit le déclarer dans `web.xml`
  - soit l'annoter par `@WebFilter('/chemin')`
- ayant comme rôle d'intercepter des requêtes sur une servlet
  - Si un filtre est déclaré sur la route `/FirstServlet`
  - Avant que le `doGet()` ou le `doPost` de la servlet soit exécuté, c'est le `doFilter()` du filtre qui sera exécuté
- le chaînage de filtres est possible

Filters

Contenu de `FirstServlet`

```
@WebServlet("/FirstServlet")
public class FirstServlet extends HttpServlet {
        private static final long serialVersionUID = 1L;

        protected void doGet(HttpServletRequest request,
           HttpServletResponse response) throws ServletException,
           IOException {

                this.getServletContext().getRequestDispatcher("/WEB-INF
                   /vueFirst.jsp").forward(request, response);
        }

        protected void doPost(HttpServletRequest request,
           HttpServletResponse response) throws ServletException,
           IOException {
                // TODO Auto-generated method stub
                doGet(request, response);
        }

}
```

Filters

Contenu de `vueFirst.jsp`

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
   pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Add Person Form</title>
  </head>
  <body>
    <form action="SecondServlet">
      Nom : <input type=text name=nom> <br/>
      Prenom : <input type=text name=prenom> <br/>
      Age : <input type=text name=age min=0 max=150> <br/>
      <button type=submit> Add </button>
    </form>
  </body>
</html>
```

Filters

Contenu de `SecondServlet`

```
@WebServlet("/SecondServlet")
public class SecondServlet extends HttpServlet {
        private static final long serialVersionUID = 1L;

        protected void doGet(HttpServletRequest request,
           HttpServletResponse response) throws ServletException,
           IOException {

                this.getServletContext().getRequestDispatcher("/WEB-INF
                   /vueSecond.jsp").forward(request, response);
        }

        protected void doPost(HttpServletRequest request,
           HttpServletResponse response) throws ServletException,
           IOException {
                // TODO Auto-generated method stub
                doGet(request, response);
        }

}
```

Filters

Contenu de `vueSecond.jsp`

```
<%@ page language="java" contentType="text/html;
   charset=UTF-8" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/
       html; charset=UTF-8">
    <title>Confirmation</title>
  </head>
  <body>
    Bonjour ${ prenom } ${ nom }.
  </body>
</html>
```

Filters

Hypothèse

- supposant qu'on veut filtrer l'ajout de personne avec une valeur pour l'attribut `age` négative ou supérieure à 150

Solution : créer un filtre

- Faire un clic droit sur `src` et aller à `New > Filter`
- Saisir `org.eclipse.filter` comme `Package name`
- Saisir `PersonneFilter` comme `Class name`
- Cliquer sur `Next` et remplacer l'`URL Pattern` par `/SecondServlet` (la route de la deuxième servlet) puis valider

Filters

Contenu de `PersonneFilter`

```
@WebFilter("/SecondServlet")
public class PersonneFilter implements Filter {

  public void destroy() {
    // TODO Auto-generated method stub
  }

  public void doFilter(ServletRequest request, ServletResponse response
     , FilterChain chain) throws IOException, ServletException {

    chain.doFilter(request, response);
  }
  public void init(FilterConfig fConfig) throws ServletException {
        // TODO Auto-generated method stub
  }
}
```

Remarque

`chain.doFilter(request, response);` appelle le filtre suivant (s'il en existe), sinon la servlet ayant la route `/SecondServlet` sera exécutée.

Filters

Pour s'assurer que le filtre est exécuté avant la servlet, on modifie le contenu de `PersonneFilter`

```
@WebFilter("/SecondServlet")
public class PersonneFilter implements Filter {

  public void destroy() {
    // TODO Auto-generated method stub
  }

  public void doFilter(ServletRequest request, ServletResponse response
     , FilterChain chain) throws IOException, ServletException {
    System.out.println("In filter");
    chain.doFilter(request, response);
  }
  public void init(FilterConfig fConfig) throws ServletException {
        // TODO Auto-generated method stub
  }
}
```

Filters

On modifie aussi le `doGet()` de `SecondServlet`

```
@WebServlet("/SecondServlet")
public class SecondServlet extends HttpServlet {
  private static final long serialVersionUID = 1L;

  protected void doGet(HttpServletRequest request, HttpServletResponse
     response) throws ServletException, IOException {
    System.out.println("In SecondServlet");
    this.getServletContext().getRequestDispatcher("/WEB-INF/vueSecond.
       jsp").forward(request, response);
  }

  protected void doPost(HttpServletRequest request, HttpServletResponse
      response) throws ServletException, IOException {
    // TODO Auto-generated method stub
    doGet(request, response);
  }
}
```

Remarque

On lance l'application, on remplit les trois champs et on clique sur le bouton $\Rightarrow$ dans la console, `In filter` est affiché en premier ensuite `In SecondServlet`.

Filters

Filtrons maintenant les valeurs de l'attribut `age`

```
@WebFilter("/SecondServlet")
public class PersonneFilter implements Filter {

  public void destroy() {
    // TODO Auto-generated method stub
  }

  public void doFilter(ServletRequest request, ServletResponse response
     , FilterChain chain) throws IOException, ServletException {
    int age = Integer.parseInt(request.getParameter("age"));
    System.out.println("age------" + age);
    if (age >= 0 && age <150)
      chain.doFilter(request, response);
    else
      response.getWriter().println("corriger l'age");
  }
  public void init(FilterConfig fConfig) throws ServletException {
        // TODO Auto-generated method stub
  }
}
```