Desarrollo de Aplicaciones Web

Servlets. Inicialización.

USC UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

Pablo García Tahoces

Parámetros de inicialización

Parámetro.- dato de tipo *string* , que permite almacenar información utilizada en la aplicación correspondiente.

Resuelve el inconveniente de tener que introducir dicha información como parte del código a compilar.

Ej.- PrintWriter out= response.getWriter();
out.println("mrpablo@usc.es");

Parámetros de inicialización

Solución mediante uso de parámetros del propio servlet

```
<servlet>
    <servlet-name> ……. <servlet-name>
    <servlet-class> ……. </servlet-class>

    <init-param>
            <param-name> miEmail </param-name>
            <param-value> miEmail@usc.es </param-value>
   <init-param>
    ….
    ….
</servlet>
```

```
out.println(getServletConfig().getInitParameter("miEmail"));
```

Parámetros de contexto

Parámetros análogos a los de inicialización, pero disponibles para la aplicación completa.

```
<web-app>
    <servlet>
        <servlet-name> ……. <servlet-name>
        <servlet-class> ……. </servlet-class>
    </servlet>

    <context-param>
        <param-name> miEmail </param-name>
        <param-value> miEmail@usc.es </param-value>
    <context-param>
</web-app>
```

```
ServletContext context= getServletContext();
out.println(context.getInitParameter("miEmail"));
```

Atributos

Atributo.- objeto que consta de nombre y valor. Además tiene un periodo de validez (vida) determinado.

Pueden manejarse en términos de contexto, solicitud o sesión, existiendo métodos adecuados para las interfaces de cada caso respectivo: *ServletContext*, *ServletRequest* y *HttpSession*. Estos métodos son:

- *getAttributeNames();*
- *getAttibute(String);*
- *setAttribute(String);*
- *removeAttribute(String);*

Atributos

Ej.-

```
public void doGet(HttpServletRequest, …..) …{

    response.setContentType(“text/html”);

    PrintWriter out= response.getWriter();
    out.println(“Testeo de atributos de contexto<br>”);

    getServletContext().setAttribute(“PBI”, “6”);
    getServletContext().setAttribute(“DAAI”, “7,5”);

    out.println(getServletContext().getAttribute(“PBI”);
    out.println(getServletContext().getAttribute(“DAAI”);
}
```

Atributos. Sincronización.

Si se quiere proteger el valor del atributo de la acción de otras hebras, la solución pasa por cortar el acceso de las mismas vía sincronización.

Ej.-

```
public synchronized void doGet(HttpServletRequest, …..) … {
….
….
 }
```

Atributos. Sincronización.

Sincronización a nivel de contexto.- permite cortar el acceso al resto de *servlets* de la *webapp*, siempre y cuando el resto de *servlets* se sincronicen también..

Ej.-

```
public void doGet(HttpServletRequest, …..) …{

    response.setContentType("text/html");

    PrintWriter out= response.getWriter();
    out.println("Testeo de atributos de contexto<br>");

    synchronized(getServletContext()){
        getServletContext().setAttribute("PBI", "6");
        getServletContext().setAttribute("DAAI", "7,5");

        out.println(getServletContext().getAttribute("PBI");
        out.println(getServletContext().getAttribute("DAAI");
    }
}
```

Atributos. Sincronización.

Sincronización a nivel de sesión.- permite cortar el acceso al resto de *servlets* de la *sesión*, siempre y cuando el resto de *servlets* se sincronicen también..

Ej.-

```
public void doGet(HttpServletRequest, …..) …{

    response.setContentType("text/html");

    PrintWriter out= response.getWriter();
    out.println("Testeo de atributos de contexto<br>");

    synchronized(session){
        getServletContext().setAttribute("PBI", "6");
        getServletContext().setAttribute("DAAI", "7,5");

        out.println(getServletContext().getAttribute("PBI");
        out.println(getServletContext().getAttribute("DAAI");
    }
}
```