Universidad de Oviedo

EPI GIJÓN

Universidad de Oviedo

Servlets

Enol García González
Universidad de Oviedo
9 de septiembre de 2023

Contenidos

Servidores de aplicaciones

Definición

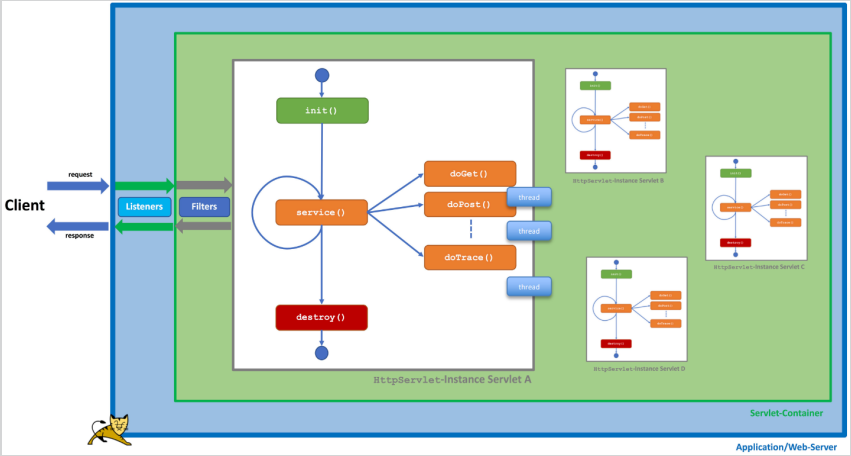
Programa que provee la infraestructura necesaria para aplicaciones web empresariales

- Los programadores van a poder dedicarse casi en exclusiva a implementar la lógica del dominio
- Servicios como seguridad, persistencia, transacciones, etc. son proporcionados por el propio servidor de aplicaciones

Servlets

Un servlet es un programa Java que acepta peticiones HTTP y genera respuestas utilizando también el protocolo HTTP.
Principalmente se usan para la construcción de páginas web dinámicas.
En Java se crea extendiendo la clase HTTPServlet, que forma parte de JEE.
Los servlets se ejecutan dentro de un contenedor de Servlets que está dentro de un servidor web.

Servlets – Ciclo de vida



Servlets – Ejemplo de un Hola Mundo

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

public class HolaMundoServlet extends HttpServlet {
  public void doGet(HttpServletRequest request, HttpServletResponse response)
                        throws ServletException, IOException {
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    out.println("<html><head>");
    out.println("<title>Servlet Hola Mundo </title>");
    out.println("</head><body>");
    out.println("<h1>Hola Mundo!</h1>");
    out.println("</body></html>");
  }
}
```

Servlets – Registro

Con el fichero `web.xml`

```
<servlet>
  <servlet-name>HolaMundo</servlet-name>
  <servlet- class>uo.sdi.servlet.
      HolaMundoServlet</servlet-class>
</servlet>

<servlet-mapping>
  <servlet-name>HolaMundo</servlet- name>
  <url-pattern>/HolaMundoCordial</url-
      pattern>
</servlet-mapping>
```

Con una anotación

```
@WebServlet(name = "HolaMundo",
    urlPatterns = {"/HolaMundoCordial"})
public class HolaMundoServlet extends
    HttpServlet {

/** ... */

}
```

Servlets – Parámetros

```
<form action="http://server/app/HelloWorld" method="POST">
  Nombre: <input type="text" name="NombreUsuario" />
  <input type="submit" value="Aceptar" />
</form>
```

```
public class HelloWorld extends HttpServlet {
  protected void doGet(HttpServletRequest request, HttpServletResponse response)
      throws ServletException, IOException {
    String nombre = (String) request.getParameter("NombreUsuario")

    /** ... */
  }
}
```

Servlets – Ámbitos de las variables

Request

- Sólo vigente mientras se realiza la petición
- Sólo el Servlet que procesa la petición ve los datos

Sesión

- Vigente mientras no se destruya la sesión
- Sólo el Servlet que procesa la petición ve los datos

Contexto

- Vigente durante toda la ejecución del proyecto
- Todos los servlets comparten su información

Servlets – Sesión

Uno de los mayores problemas del protocolo HTTP es que no mantiene el estado.
Complica guardar las acciones del usuario.
Posibles soluciones:

- Información en la URL
- Campos ocultos
- Cookies

Servlets – HttpSession

Al trabajar con Servlets tenemos una solución técnica que nos facilita la gestión de la sesión: El objeto **HttpSession**

- HttpSession almacena objetos en el lado del servidor
- Cada usuario tendrá asociada un objeto HttpSession con un identificador único
- En el cliente se crea una cookie con el identificador de la sesión
- En cada petición, el cliente incluye su identificador de sesión

Servlets – Uso de HttpSession

Al desarrollar los servlets podemos utilizar el objeto HttpSession con el método `HttpServletRe`
El valor true hará que la sesión se cree en caso de no existir.
La sesión se cerrara automáticamente después de un tiempo sin peticiones o al llamar al método *HttpSession.invalidate()*.

Servlets – Uso de HttpSession

Los dos método más importantes, que nos permiten gestionar la inforamción guardada en la sesión son:

- `setAttribute` para registrar un atributo o modificar su valor
- `getAttribute` para recuparar su valor

Servlets – Contexto

Es común a todos los Servlets, nos permite compartir información y objetos entre los distintos usuarios.
Se accede con el objeto ServletContext. Sus dos métodos más importantes son:

- `setAttribute` para registrar un atributo o modificar su valor
- `getAttribute` para recuparar su valor

Servlets – Ejemplo Contador

```
public class ContadorServlet extends HttpServlet {
  protected void doGet(HttpServletRequest request, HttpServletResponse response)
      throws ServletException, IOException {
    ServletContext contexto=request.getServletContext();

    Integer contador = (Integer) contexto.getAttribute("contador");
    if (contador == null)  contador = 0;
    contador++;
    contexto.setAttribute("contador", contador);
  }
}
```

Servlets – Vida de las variables