élogos

# DESARROLLO DE APLICACIONES WEB CON SERVLETS

élogos

# Redireccionamiento y transferencia de peticiones

## TRANSFERENCIA DE UNA PETICIÓN

En este capítulo te mostraremos cómo se transfiere una petición desde un servlet a otro recurso de la aplicación.

### Trasferencia y redireccionamiento

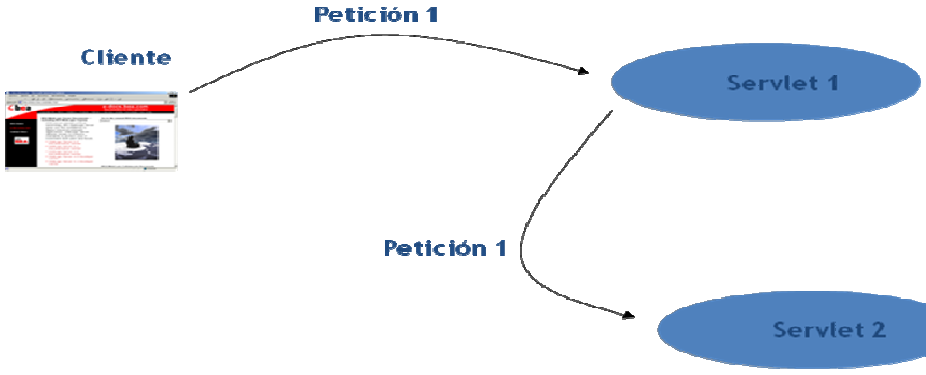
Hay muchas situaciones en las que un Servlet no puede procesar por si solo una determinada petición, por lo que necesita apoyarse en otros Servlets, páginas JSP o HTML para completar las operaciones a realizar.
En función de si el recurso de apoyo se encuentra en la misma o en otra aplicación Web, el Servlet puede utilizar uno de los siguientes mecanismos para poder hacer uso del mismo:

- Transferencia de una petición: el Servlet transfiere la petición a otro componente de la misma aplicación.
- Redireccionamiento: el Servlet hace un llamamiento a un componente que se encuentra en otra aplicación Web.

### Trasferir una petición

Cuando la petición que llega a un determinado Servlet debe ser atendida por otro servlet o JSP de la misma aplicación, recurriremos al reenvío de peticiones.

El reenvío consiste en derivar la misma petición a otro componente de la aplicación, compartiendo ambos los mismos objetos **request** y **response**, por lo que los dos tendrán acceso a los datos enviados por el cliente.



### Objeto RequestDispatcher

La transferencia de una petición ser realiza a través del objeto RequestDispatcher, interfaz esta que se encuentra definida en el paquete javax.servlet.

Para obtener un objeto RequestDispatcher apuntando al servlet destino desde el Servlet origen utilizaremos el método getRequestDispatcher() definido en la interfaz HttpServletRequest, y cuyo formato es:

```
RequestDispatcher getRequestDispatcher(String url)
```

Donde la url representa la dirección del recurso destino, relativa a la aplicación Web.

Para obtener un RequestDispatcher asociado al servlet2 desde el método service() del servlet1 sería:

***RequestDispatcher rd = request.getRequestDispatcher("/servlet2");***

Obsérvese como la dirección del recurso debe ir precedida por la barra inclinada '/'.

Se puede obtener también un RequestDispatcher que apunte a una página HTML: getRequestDispatcher("/mipagina.html")

### Métodos de RequestDispatcher

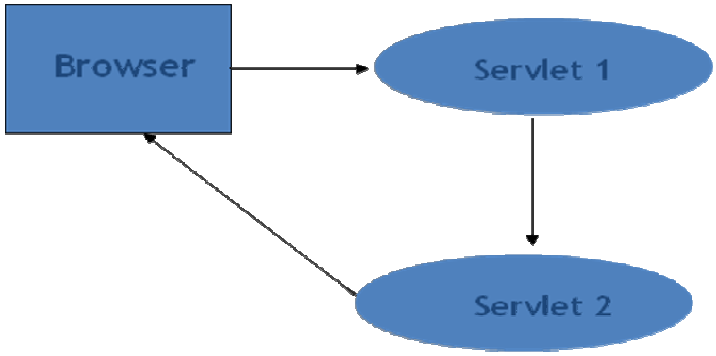
Una vez obtenido el RequestDispatcher, podremos llamar a alguno de sus dos métodos para realizar la transferencia de la petición. Estos métodos son:

- **forward(HttpServletRequest request, HttpServletResponse response).** El Servlet pasa el control de forma permanente al nuevo Servlet , el cual se encarga de enviar la respuesta al cliente.

- **include(HttpServletRequest request, HttpServletResponse response).** El control se pasa de forma temporal al nuevo Servlet, de modo que cuando termina la ejecución de su método service() devuelve el control al primer Servlet.

Tanto en un caso como en otro, los objetos request y response son compartidos por ambos servlets.

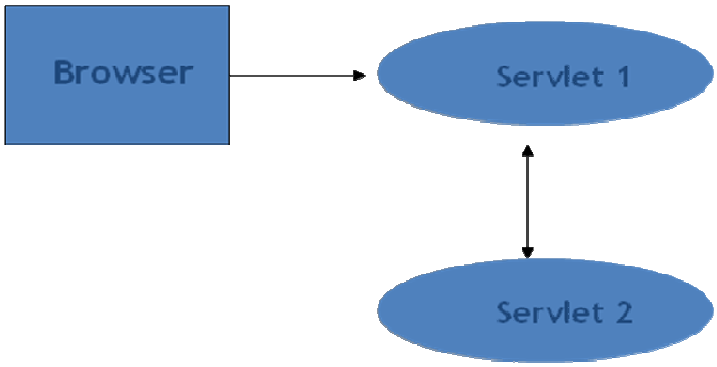
**Método forward**




Se utilizará en aquellas situaciones en las que el servlet destinatario tenga que encargarse de generar completamente las respuesta del cliente:

```
RequestDispatcher rd = request.getRequestDispatcher("/servlet2");
rd.forward(request, response);
```

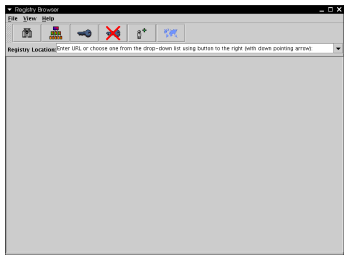
**Método include**




Si después de ejecutarse el servlet destinatario se debe devolver el control al servlet origen para que complete la respuesta, utilizaremos el método include():

```
RequestDispatcher rd = request.getRequestDispatcher("/servlet2");
rd.include(request, response);
```

## REDIRECCIONAMIENTO

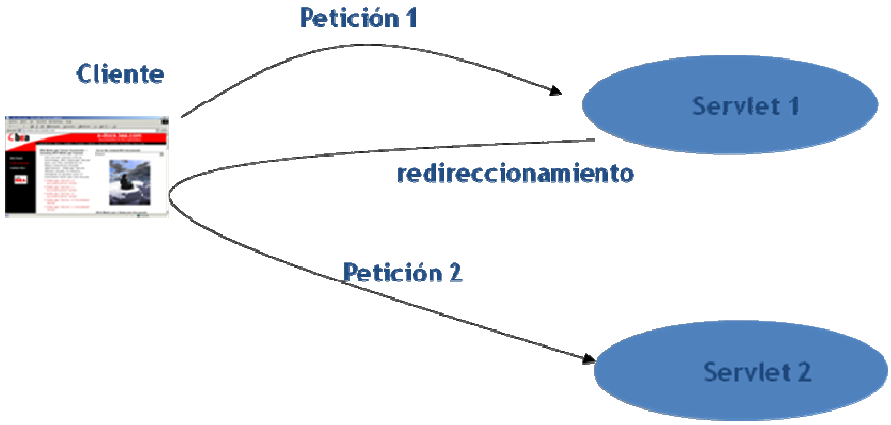


En este capítulo te mostraremos cómo redireccionar al usuario desde un servlet a otra URL.

### Funcionamiento

Como su nombre indica, por redireccionamiento entendemos redireccionar al usuario desde un servlet a otra URL diferente.

Mediante el redireccionamiento lo que hacemos es forzar al navegador a realizar una petición a otra dirección diferente, todo ello de forma transparente para el usuario, que lo único que aprecia es que la dirección indicada en su navegador ha cambiado y apunta a otro lugar diferente.




### El objeto HttpServletResponse

Para realizar un redireccionamiento a otra URL desde un Servlet, utilizaremos el método **sendRedirect()** del objeto response. El formato de este método es el siguiente:

*void sendRedirect(String url)*

Donde url representa la dirección absoluta del recurso al que queremos enviar al usuario:

*response.sendRedirect("http://java.sun.com");* //redireccionamiento a página de inicio de Java

Al ejecutar la instrucción anterior, se envia una respuesta al navegador cliente con código de estado 302 y cabecera "location" con el valor de la nueva URL, lo que provoca que el navegador se dirija automáticamente a esta dirección.

**Pérdida de parámetros**

Una de limitaciones del redireccionamiento es que, al tratarse de peticiones diferentes, los parámetros que llegan al Servlet no están accesibles al recurso destino.

Si hubiera una necesidad de compartir datos entre ambos recursos, deberíamos recurrir a la inserción de datos en la URL:

**Servlet 1**

```
String n = request.getParameter("nombre");
response.sendRedirect("servlet2?name=n);
```

**Servlet 2**

```
String n = request.getParameter("name");
out.prinln("Su nombre es: "+n);
```