Ejercicios de Comunicación

Índice

1. Ejemplo de contexto

Vamos a probar la aplicación `jsp-sesion05-contexto` incluida en los ejercicios de la sesión.

a) Desplegar la aplicación web en Tomcat, y acceder a la dirección:

```
http://localhost:8080/jsp-sesion05-contexto
```

Veremos la aplicación web ya en marcha.

b) La aplicación web nos permite visualizar los atributos de contexto definidos y sus valores, y añadir nuevos atributos. A parte de los atributos que nosotros añadimos manualmente, ¿hay más atributos de contexto definidos?

c) Podemos añadir nuevos atributos de contexto. Daremos un nombre del atributo, y un texto que contendrá como valor. Además como valor también se introducirá el identificador de sesión del navegador que haya creado dicho atributo. Abrir distintos navegadores y añadir atributos de contexto desde cada uno. Comprobar que en cada navegador vemos tanto los atributos creados en su sesión, como lo atributos creados creados en las sesiones de otros navegadores (el identificador de sesión será distinto).

d) Si nos fijamos en el descriptor de despligue, `web.xml`, veremos que se ha añadido un listener sobre los atributos del contexto. Este listener imprime mensajes en el *log* indicando cuando se añade, elimina o reemplaza un atributo de contexto. Comprobar en el fichero de logs correspondiente que se han registrado los cambios en los atributos que hayamos hecho.

2. Chat con servlets

Vamos a realizar una aplicación de chat utilizando servlets. En el directorio `jsp-sesion05-chat` de los fuentes de la sesión podrás encontrar la base sobre la que construiremos el chat. Cada mensaje de chat estará encapsulado en la clase `Mensaje`, y la lista de mensajes actual del chat se encontrará en la clase `ColaMensajes`.

Además se proporcionan los ficheros HTML necesarios para la aplicación. El fichero `index.html` contiene el formulario de login para que un usuario introduzca el *nick* con el que entrará en el chat (no se solicita ningún password para validar). El login se hará efectivo por el servlet `LoginUsuarioServlet` también proporcionado, que introducirá el *nick* del usuario en la información de sesión y nos redirigirá al chat. En el subdirectorio `chat` tendremos los ficheros estáticos del chat:

`chatFrames.htm`	Página principal de los frames de la aplicación chat. Mostrará un frame con el

	formulario para enviar mensajes, y otro con la lista de mensajes enviados.
`enviaMensaje.h`	Formulario para enviar mensajes al chat.
`error.html`	Página que nos muestra un mensaje de error cuando se intenta enviar un mensaje sin haber hecho *login*.
`cabecera.html`	Cabecera de la tabla de mensajes, a incluir al comienzo de la página de lista de mensajes.
`pie.html`	Pie de la tabla de mensajes, a incluir al final de la página de lista de mensajes.

Ahora deberemos implementar los servlets para el envio de mensajes y para la consulta de la lista de mensajes enviados. Se pide:

a) La cola de mensajes será el objeto común al que acceden los servlets para el envio y la consulta de estos mensajes. Por lo tanto el objeto deberá añadirse como atributo del contexto. Esto lo tendremos que hacer antes de que cualquier servlet se haya ejecutado. Para ello debemos crear un objeto `ServletContextListener` que en la creación del contexto inicialice la cola de mensajes (`ColaMensajes`) y la introduzca como atributo en el contexto global (atributo `es.ua.jtech.jsp.sesion5.chat.mensajes`).

b) Una vez tenemos creada la cola de mensajes, deberemos implementar el servlet `EnviaMensajeServlet`, que tome un mensaje como parámetro (el nombre del parámetro es `texto`), y lo añada a la lista de mensajes con el nick del usuario actual (obtenido del atributo `es.ua.jtech.jsp.sesion5.chat.nick` de la sesión). Una vez enviado el mensaje, mostraremos en la salida el contenido de `enviaMensaje.html`, mediante un objeto `RequestDispatcher`. Si no hubiese ningún usuario en la sesión, no se registrará el mensaje y se deberá redirigir la salida a `error.html`

c) Por último, deberemos implementar el servlet `ListaMensajesServlet` que mostrará todos los mensajes del chat. Este servlet debe:

- Para que la lista de mensajes se actualice periodicamente en el cliente, haremos que se recargue cada 5 segundos. Añadir la cabecera HTTP correspondiente a la respuesta para que esto ocurra (cabecera `Refresh`, indicando como valor el número de segundos que tardará en recargar).
- Incluir el contenido del fichero estático `cabecera.html` al comienzo del documento generado, y `pie.html` al final, para enmarcar la zona donde aparecen los mensajes del chat.

Nota

Se debe obtener el `PrintWriter` para escribir la respuesta antes de hacer el primer `include`, ya que de lo contrario se obtendría una excepción de tipo `IllegalStateException`. Esto de debido a que el `include` ya habría obtenido previamente el flujo de salida de la respuesta.

- Obtener el *nick* del usuario actual de la sesión. Los mensajes enviados con este *nick* se mostrarán en negrita, el resto se mostrarán de forma normal.

d) Comprobar que el chat funciona correctamente. Conectar desde varios clientes a un mismo servidor.