1º Práctica, Manejo de Cookies y Sesiones J2EE

Introducción

El objectivo de ésta primera práctica consiste en manejar la persistencia de sesiones entre el navegador y el servidor mediante las clases Java de J2EE. Para ello se han realizado dos proyectos en Eclipse:

- **PersistentCookies**: En éste proyecto se hace uso de la clase Cookie de java para enviar al navegador una página con el número de veces que lleva visitada dicha página. Adémas se guarda en fichero un mapa clave-valor que permite guardar en disco las sesiones de los usuarios.
- **PersistenciaSesiones**: Para este otro proyecto hemos usado la clase nativa de Java HttpSession que permite almacenar las sesiones del usuario de forma transparente para el programador.

PersistentCookies

Dado que los requisitos para esta parte de la práctica son muy sencillos, sólo he implementado tres clases:

1. **Utils**: clase de utilidades con métodos estáticos.
2. **CookieUtils**: la clase que provee CoreServlets para manejar las cookies.
3. **Contador**: clase que extiende de HttpServlet e sobreescribe el método “doGet” para atender a las peticiones de los clientes.

Flujo de la aplicación

El flujo de la aplicación es muy simple:

- Un cliente realiza una petición al servidor que escucha en el puerto correspondiente utilizando el método GET.
- El servidor recibe la petición e intenta leer la cookie “contador” del cliente, si existe le suma 1, si no existe la inicializa a 0 y le suma 1 igualmente.
- El servidor inserta o actualiza en el fichero de sesiones una entrada de tipo “IP ----> contador”.
- El servidor crea una nueva cookie en la respuesta con el nuevo valor de contador y envía la respuesta al cliente.

Consideraciones

El fichero de propiedades de sesiones se crea en la carpeta **home** del usuario con el nombre session.properties. Este fichero se carga en memoria con el objeto properties de Java que es en esencia un HashMap clave-valor. Para no perder los datos del usuario en caso de que exista un apagón o algo similar el fichero se sobreescribe con cada nueva petición al servidor.

PersistenciaSesiones

Los requisitos para este proyecto son un poco más complejos que en el anterior, por lo que se ha realizado una distribución mediante paquetes de Java con el siguiente esquema:

- ***model***
  - **Usuario:** POJO que representa un usuario en el sistema.
- ***servlets***
  - **Contador:** Actua como mock para demostrar que se ha realizado el login correctamente y que la sesión del usuario es efectivamente persistente en el servidor y se envía correctamente al usuario.
  - **Login**: Permite autenticar los datos de entrada del Usuario que se envían al servidor mediante el método POST.
  - **Logout**: Invalida la sesion y redirige a la página de error del sistema.
  - **NoSessionException**
- ***utils***
  - **CookieUtilities**
- ***validation***
  - **InvalidUserException**
  - **Validator**: Clase mock que valida si el los datos del usuario introducidos son correctos y están en el sistema. Con fines prácticos los datos de prueba son **demo** para el usuario y **demo** para la contraseña.

Flujo de la aplicación

El flujo de la aplicación es el siguiente:

- El usuario entra en la aplicación la cual le presenta un formulario inicial para que éste introduzca sus datos.
- El formulario envía la petición POST al servlet Login que corroborá dichos datos mediante el validador. Si los datos son correctos el servidor prepara la sesión para el usuario y le envía una página de bienvenida customizada para su sesión, en caso contrario notifica con una página que contiene un enlace para volver al comienzo.
- Una vez autenticado el usuario la aplicación es basicamente la misma que la anterior, la diferencia reside en que esta vez utilizamos la sesión del usuario para guardar el contador usando las cookies solamente para guardar el JSESIONID, que se gestiona automaticamente por la clase HttpSession.
- Finalmente el usuario puede hacer click en el enalce de logout el cual le redirige al servlet correspondiente e invalida la sessión redirigiendo a la página de incio y comenzando así de nuevo el ciclo.

Consideraciones

Por motivos de seguridad, en el servlet Contador se comprueba que hay una sesión, lanzando en caso negativo una excepción controlada y redirigiendo a la página de inicio.