Ejercicios de procesamiento de peticiones

Índice

1. Recogida de parámetros del usuario

La aplicación `cw-sesion02` contiene un formulario `form_datos.html` con una serie de campos (tipo texto, listas, checkboxes...). Se pide que dicho formulario, en su action, llame al servlet `es.ua.jtech.cw.sesion02.ejercicios.DatosServlet` que deberéis crear e implementar. El servlet recogerá todos los parámetros del formulario y los mostrará en una tabla de dos columnas (una con el nombre del parámetro y otra con el valor).

2. Trabajando con redirecciones

Sobre la aplicación anterior, tenemos otro formulario `form_datos2.html` idéntico al del ejercicio anterior, que deberá llamar al servlet `es.ua.jtech.cw.sesion02.ejercicios.DatosServlet2`. Crear este servlet y hacer que redirija a la página `bienvenida.html` con un mensaje de bienvenida, si los datos introducidos en el formulario son correctos, y a la misma página `form_datos2.html` si hay algún dato incorrecto. Entenderemos por dato incorrecto que alguno de los campos de texto se quede vacío. Utilizad el método `sendRedirect` de la respuesta para las redirecciones.

3. (*)Un buscador sencillo

En el fichero `libros.txt` hay un listado de libros, indicando su ISBN, título y autor. Cread e implementad un servlet `es.ua.jtech.cw.sesion02.ejercicios.LibroServlet` que lea dicho fichero, guarde los libros en un `ArrayList`, y reciba un parámetro `cadena`. Como resultado, sacará todos los libros de la lista que contengan dicha cadena (en el título, en el autor, o en cualquier parte de la cadena). Podéis hacer también una página HTML `libros.html` con el formulario de búsqueda que llame al servlet, para poderlo probar

4. Distinguir el navegador

Muchas veces, cuando escribamos una aplicación Web, nos va a interesar poder distinguir el tipo de navegador que está utilizando el cliente, porque en función del mismo se podrán hacer o no determinadas acciones. Por ejemplo, el código Javascript que se emplea en un navegador Internet Explorer es diferente a veces del que se emplea en uno Netscape. Para probar a distinguir entre navegadores, vamos a crear el servlet `es.ua.jtech.cw.sesion02.ejercicios.NavServlet` para que identifique si el cliente accede desde un tipo de navegador u otro. Para ello leemos la cabecera `User-Agent` con el método `getHeader(...)` de la petición, y comprobamos su valor. Mostrad la cadena

en una página, y cargad dicha página desde dos o tres navegadores diferentes. Cada uno mostrará algún rasgo distintivo en dicha cadena, que lo identifique de los demás. Una vez tengáis una parte de texto que los diferencia (por ejemplo, imaginemos que en Netscape el `User-Agent` tiene la cadena "Netscape", y en el otro navegador (Konqueror, por ejemplo), no la tiene) haríamos con algo como:

```
public void doGet(HttpServletRequest req, ...) throws ...
{
        String nav = req.getHeader("User-Agent");
        // Cambiar "Netscape" por el texto que sea
        if (nav.indexOf("Netscape") != -1)
                ... // Netscape
        else
                ... // Otro navegador (Konqueror)
        ...
}
```

Una vez distinguido el navegador, ya se podría hacer algo que sólo sirviera para ese navegador. En este caso, por simplificar, vamos a cargar como imagen el logo del navegador que hayamos detectado. Tenéis en la plantilla las imágenes GIF correspondientes a tres navegadores o entornos diferentes. Colocad como imagen de la página la del navegador que hayáis detectado (con una etiqueta `<img>` de HTML).

5. Redirecciones con retardo

El formulario `form_datos3.html` se envía al servlet `es.ua.jtech.cw.sesion02.ejercicios.CompruebaServlet`. Se pide crear e implementar dicho servlet para comprobar que los datos sean correctos (que no haya ningún campo de texto vacío). En el caso de que no haya errores el servlet simplemente mostrará un mensaje indicando que todo ha ido bien. Si hay algún error, el servlet debe redirigir al servlet `es.ua.jtech.cw.sesion02.ejercicios.ErrorCompruebaServlet`, que deberéis crear para que muestre un mensaje con el error producido (indicando qué campo está incompleto), y a los 5 segundos redirija al formulario anterior (utilizando una cabecera de respuesta `Refresh`).

6. (*) Loggear variables CGI

Sobre el servlet `es.ua.jtech.cw.sesion02.ejercicios.CompruebaServlet` anterior vamos a añadir mensajes de log de tipo INFO, para que:

- Tras cada petición (por `doGet` o `doPost`), genere un mensaje de tipo INFO que indique la IP del cliente que solicitó la petición (variable CGI `REMOTE_ADDR`), el tipo de petición (variable CGI REQUEST_METHOD), y el tipo de navegador (cabecera `User-Agent`).
- El mensaje en conjunto deberá tener el formato siguiente:

  `día/mes/año hora:minuto:segundo - texto del mensaje - nueva línea`

Haced que los mensajes de log se guarden en un fichero `visitas.txt`, en la carpeta que prefiráis. Para ello utilizad un fichero `log4j.properties` colocado en `WEB-INF/classes`.