

Ejercicios sesión 1 - Introducción a Struts

Índice

1 Instalación de la aplicación de ejemplo.....	2
2 Implementar un caso de uso completo (1 punto).....	2
3 Gestionar errores en las acciones (1 punto).....	3
4 Seguridad declarativa (1 punto).....	3

1. Instalación de la aplicación de ejemplo

Durante las cuatro sesiones de Struts realizaremos los ejercicios sobre la misma aplicación de ejemplo. Se trata de una sencilla aplicación para almacenar las tareas pendientes (ToDo) de un usuario. Hay que tener en cuenta que es simplemente una aplicación de ejemplo. El objetivo es tener una aplicación para poder "trastear" con Struts más compleja que un "hola mundo" pero no tanto como una aplicación real. Tanto la arquitectura de la aplicación como el interfaz de usuario se deberían mejorar en una implementación más realista.

Los casos de uso de la aplicación son:

- Hacer login
- Ver la lista de tareas pendientes
- Crear una nueva tarea
- Eliminar una tarea existente

Para instalar la aplicación, seguid estos pasos:

1. Bajáos el proyecto de Eclipse e importadlo a vuestro espacio de trabajo
2. **Crear la base de datos:** en la carpeta database hay un script SQL para crear la base de datos de MySQL. Podéis crearla con las herramientas gráficas de MySQL o bien tecleando desde una terminal

```
mysql -u root -p < todo.sql (os pedirá el password: especialista)
```

3. Para asegurarnos de que la conexión con la base de datos funciona, haced login con el usuario "struts" y el password "mola". Debe aparecer una lista de tareas pendientes.

2. Implementar un caso de uso completo (1 punto)

Como mejor se ve el funcionamiento de Struts es siguiendo el flujo de ejecución de un caso de uso completo. Para ello vais a implementarlo vosotros y luego probarlo. El caso de uso en cuestión es el de mostrar todos los datos de una tarea. En la página que lista todas las tareas hay un enlace "Ver detalles" al lado de cada una, que apunta a "verTarea.do". Falta por implementar:

- La clase Java con la acción
- El JSP que muestre el resultado (verTarea.jsp)
- El mapeo en el struts-config.xml entre la acción y los posibles resultados. En este caso habrá un único resultado posible, "OK".

En la plantilla, la clase es `ua.jtech.struts.acciones.VerTareaAccion` devuelve simplemente el forward "OK". El JSP "verTarea.jsp" muestra simplemente un mensaje de saludo. Se recomienda que probéis primero a hacer el mapeo en el struts-config.xml, comprobéis que funciona y luego hagáis la implementación real de la acción y del JSP.

Veámoslo con un poco más de detalle:

1. En cuanto a **La clase Java VerTareaAccion**.
 - El único posible resultado (*forward*) por el momento será "OK".
 - El acceso a la base de datos nos lo da la clase TareaDAO. Para obtener una instancia, llamar a `TareaDAO.getInstance()`. Para obtener los datos de una tarea, llamar a `getTarea(id_de_la_tarea)`. El id de la tarea lo podemos sacar del parámetro HTTP "id".
 - Una vez se obtenga un objeto Tarea con todos los datos, colocarlo en el request para que se pueda mostrar desde el JSP.
2. En cuanto a **La vista verTarea.jsp**: Usad el lenguaje de expresiones de JSP para simplificar la sintaxis, podéis tomar como ejemplo `tareas.jsp`.

3. Gestionar errores en las acciones (1 punto)

Añadir gestión de errores a la acción `VerTarea`, de modo que si se solicita el id de una tarea no existente, se detecte la situación y aparezca un mensaje de error.

Pasos a seguir:

1. En el fichero `mensajes.properties` (carpeta `resources`) añadir un mensaje de error para este caso.
2. En el código de la acción:
 - Si el método `getTarea(id)` de `TareaDAO` devuelve `null`, la tarea no existe. Por tanto en este caso habrá que devolver otro resultado distinto de "OK". Llamadlo "error".
 - En caso de error, crear un `ActionMessage` asociado al mensaje del fichero `mensajes.properties` y guardarlo con `saveErrors` como se explica en los apuntes.
3. En el `struts-config.xml`, asociar el resultado "error" con la página `error.jsp`, que deberás crear.
4. En dicha página usar la etiqueta `<html:messages>` para mostrar el error.

4. Seguridad declarativa (1 punto)

La aplicación de tareas pendientes usa seguridad declarativa. Hay dos clases de usuarios (dos roles): "registrado" y "pro". Modifica la aplicación para que solo los usuarios "pro" puedan ver los detalles de la tarea.

- Modifica el `struts-config.xml` para que la acción de `VerTarea` solo se pueda ejecutar con rol "pro".
- Haz que si intenta acceder un usuario que no tenga este rol se salte a la página "sinPermiso.jsp", que deberás crear, mostrando en ella un mensaje de error apropiado.

