Ejercicios de Spring Roo

Índice

Vamos a crear una especie de prototipo reducido (pero muy reducido) del proyecto de integración. Esto nos ayudará a tener una idea aproximada de cuáles son los puntos fuertes de Roo y en qué partes de la aplicación no nos va a servir de ayuda.

En este modelo del dominio simplificado, solo tendremos tres entidades básicas: Libro, Ejemplar y Usuario (que, como en el proyecto de integración, será en realidad una clase abstracta de la que deben heredar Profesor y Alumno). Habrá una relación uno a muchos entre Usuario y Libro y otra entre Libro y Ejemplar.

Antes que nada, podéis crear una base de datos MySQL vacía para el proyecto. La llamaremos biblioteca_roo. Podemos hacerlo con mysql desde un terminal:

```
$ mysql -u root -p #(os pedirá el password)
mysql> create database biblioteca_roo;
Query OK, 1 row affected (0.00 sec)
mysql> quit
```

Cread un proyecto Roo desde STS como ya habéis hecho en la demo. Dadle de nombre jbib-roo. Como "top level package name" podéis darle es.ua.jtech.jbib, al igual que se hace en el proyecto de integración.

1. Entidades (1.5 puntos)

Como ya hemos visto, lo primero en Roo es elegir la tecnología de persistencia y la base de datos subyacente, en nuestro caso el JPA de Hibernate con MySQL:

```
roo> jpa setup --provider HIBERNATE --database MYSQL
```

Edita el fichero src/main/resources/META-INF/database.properties para cambiar la URL de conexión a la base de datos (http://localhost:3306/biblioteca_roo), el usuario (root) y el password (expertojava)

Ahora ya podemos crear las entidades. Primero vamos a crearlas vacías y luego les vamos a ir añadiendo los campos. Nótese que Usuario es abstracta y que Alumno y Profesor heredan de ella (esto no lo hemos visto en la sesión, pero como ves los comandos son relativamente sencillos)

```
roo> entity jpa --class es.ua.jtech.jbib.model.Usuario --abstract
     --inheritanceType SINGLE_TABLE
roo> entity jpa --class es.ua.jtech.jbib.model.Alumno
     --extends es.ua.jtech.jbib.model.Usuario
roo> entity jpa --class es.ua.jtech.jbib.model.Profesor
     --extends es.ua.jtech.jbib.model.Usuario
```

Ahora podemos crear las dos entidades que nos quedan: Libro y Ejemplar:

```
roo> entity jpa --class es.ua.jtech.jbib.model.Libro
roo> entity jpa --class es.ua.jtech.jbib.model.Ejemplar
```

Ya "solo" nos queda añadir los campos a las entidades. Esto puedes hacerlo editando directamente el código Java o bien con comandos de Roo, como hemos hecho en la demo y se ve en los apuntes. Hazlo como prefieras. Eso sí, no metas todos los campos que has usado en el proyecto de integración, no merece la pena. Por ejemplo los usuarios basta con que tengan login y password.

Finalmente, tendrás que establecer las relaciones uno a muchos entre Libro y Ejemplar y entre Usuario y Libro: aquí te ponemos la primera:

```
roo> focus --class es.ua.jtech.jbib.model.Libro
roo> field set --fieldName ejemplares --type Ejemplar --cardinality
ONE_TO_MANY
     --mappedBy libro
roo> focus --class Ejemplar
roo> field reference --fieldName libro --type Libro --cardinality
MANY_TO_ONE
```

2. Capa web (1,5 puntos)

Ya estamos casi preparados para crear la capa web. Pero antes tenemos que resolver un problema engorroso que ya ha surgido en la demo de las tareas: la pluralización de Roo. Cuando cree las rutas para los controller, usará pluralización anglosajona: "ejemplars", "libroes",... Una forma de solucionarlo es anotar cada clase del dominio con @RooPlural indicando cuál debe ser el plural, por ejemplo:

```
...
@RooPlural("Libros")
public class Libro {
...
```

Tendrás que anotar las cinco entidades, porque por desgracia, ninguna de ellas se "pluralizaría" correctamente con las reglas anglosajonas.

Ahora ya puedes crear la capa web

```
roo> web mvc setup
```

Es probable que Eclipse muestre un error tras este comando porque introduce nuevas dependencias en Maven. Normalmente podremos solucionarlo con Maven > Update. Ahora podemos hacer scaffolding de todos los controladores:

```
roo> web mvc all --package es.ua.jtech.jbib.web
```

Ya puedes probar el flamante interfaz web generado, como siempre con Run As>Run on server. Es evidente que aunque tenga un aspecto bastante aparente, no es lo que necesitas para implementar los casos de uso de la biblioteca. Las limitaciones de Roo se hacen más evidentes en la capa web cuando la aplicación no es un CRUD puro y duro.

Haz alguna prueba dando de alta un par de libros. Para evitar que cada vez que arranques la aplicación se cree de nuevo la base de datos y se pierdan todos los datos, ve al src/main/resources/META-INF/persistence.xml y cambia la propiedad "hibernate.hbm2ddl.auto", que ahora está en "create" a "update".

Intenta dar de alta también algún ejemplar. Si has creado los campos editando el Ejemplar.java manualmente verás que las fechas dan error porque Spring no "sabe" por defecto convertir de cadena a fecha (de hecho es bastante probable que hayas visto en el shell de Roo aparecer la recomendación de que hagas esto, ahora ya sabes por qué).. Puedes solucionarlo anotando los campos de tipo Date de Ejemplar con `@DateTimeFormat(style="M-")`. Si has creado estos campos con comandos Roo, la anotación se añade automáticamente.

Aunque hacer lo mismo que en el proyecto de integración real requiere modificar manualmente los JSP y los controller, vamos al menos a ver qué podríamos hacer para implementar el "ver ejemplares de un libro", ya que el "ver listado de libros" ya lo tenemos implementado.

Lo primero que necesitamos es crear el finder. Ejecuta el comando Roo adecuado para añadirle a la clase Ejemplar el finder findEjemplaresByLibro (tendrás que consultar los apuntes).

Una vez creado, tenemos que hacerlo accesible desde la capa web (es decir, que el controller de Ejemplar tenga un método que llame al finder creado). De nuevo tendrás que consultar los apuntes para ver cómo hacerlo.

Por desgracia, a partir de aquí Roo no nos puede ayudar demasiado. Tendríamos que modificar manualmente el listado de libros y en cada libro poner un enlace al finder, para así imitar al menos algo de lo que pasa en el proyecto de integración: al pinchar en un libro se ven los ejemplares. No es necesario que lo hagas, ya que no hemos visto en la sesión cuál es la estructura de los JSP creados.