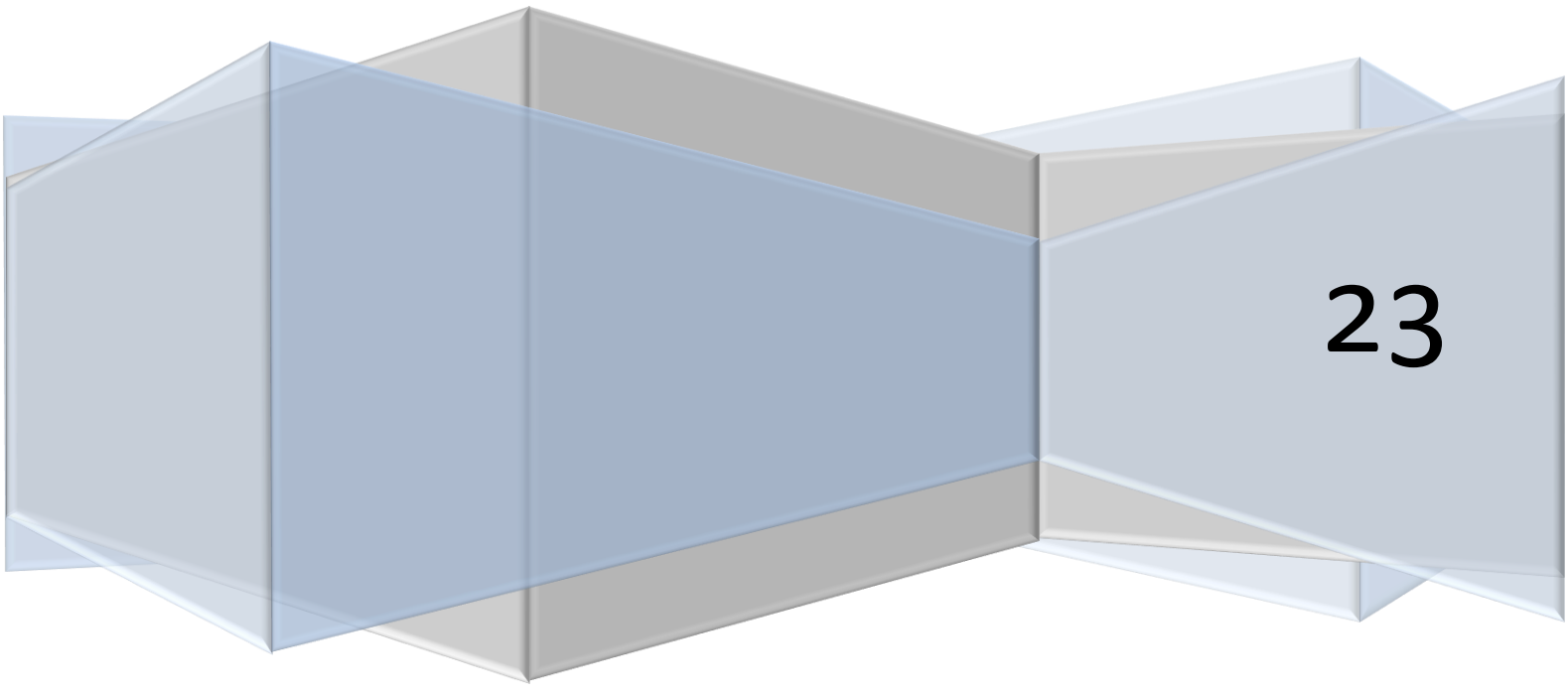
Colegio Salesianos Zaragza

Practica Sprinboot: Spring caché

Creación de una aplicación que obtiene y muestra datos de una BBDD y realiza operaciones sobre estos datos, usando spring caché para hacer un uso responsable del acceso a datos.

Luis Fernando Moreno



Introducción

La práctica consiste en crear una aplicación en SpringBoot que contenga una conexión a una BBDD. La aplicación debe poder mostrar al usuario los datos de la base de datos y realizar operaciones básicas sobre la BBDD, como añadir, editar y borrar registros. En la práctica se utilizan los fragments de thymeleaf. También se implementa Spring caché para hacer un mejor y más óptimo uso de los datos y de la conexión a la BBDD.

Desarrollo de la práctica

Para la práctica se crean varios ficheros java y html, se agrupan en varios tipos:

Entidades

Son las representaciones de las tablas en la base de datos, en Springboot se tratan como clases y objetos.

Repositorios

Clases que contienen las herramientas necesarias para comunicarse con la base de datos. Heredan de JpaRepository y son las clases que nos permite conectar y operar sobre la BBDD.

Interfaces de servicio

Estas interfaces contienen los métodos que los servicios necesitan tener

Implementación de servicios

Son clases a las que se les inyecta un repositorio, realizan operaciones sobre la base de datos como obtener datos, crear, modificar y eliminar. Extienden de las interfaces servicio.

Controladores

Son clases que contienen los controladores para dar funcionalidad a las opciones que se le muestran al usuario en la aplicación, cada controlador contiene varios métodos que se corresponden con las diferentes peticiones que puede realizar el usuario.

Layout

layout.html es una capa base que se usa como marco para dar uniformidad a las diferentes vistas de la aplicación.

Vistas (html)

Además de los explicados anteriormente, el proyecto también contiene varios ficheros html que componen las vistas que se le muestran al usuario, tenemos vistas para listar pcs, crear nuevo pc y editar pc. Existen otros ficheros html que están más relacionados a la entidad Estados, pero no se están poniendo en uso por ahora en el proyecto.

Funcionamiento:

Cuando la aplicación se inicia, carga el html listar_pcs, que nos muestra la lista de pcs. En esta misma vista tenemos opciones para añadir un nuevo pc, editar y eliminar los pcs existentes.

Si queremos añadir un nuevo pc, se lanza la petición “nuevoPc”, lo que nos lleva al método adecuado en el controlador, en este método se crea un nuevo objeto pc, y se cargan de la base de datos, (o de la caché en su caso) los estados existentes y se meten en una lista. Ambas cosas se envían al model para poder mostrar los datos en la vista.

El fichero new_Pc.html nos muestra un formulario para agregar pc, con textboxes simples. A destacar es la característica de estados, que consiste en un desplegable en el que se han cargado los estados posibles para que el usuario pueda seleccionar uno existente, y no cometer errores al elegir un estado.

Cuando se le da al botón guardar, se llama a la petición savePc, que nos lleva al método adecuado, el cual busca el estado seleccionado en la BBDD y lo inserta en el objeto Pc que se obtiene del modelo. Por último llama al servicio para guardar el pc en la BBDD y redirige a la vista de lista_pcs.html. Al listar de nuevo los pcs, se accede de nuevo a la BBDD ya que se ha modificado y por lo tanto se ha vaciado la cache. Sin embargo, si estando en la vista lista_pcs.html y se da a actualizar, ejecutando la petición, no se accede a la BBDD sino que se obtienen los datos desde la Caché.

Para la petición de editarPc, el proceso es similar a la de newPc, pero en este caso, antes de cargar la vista de edición, se obtiene de la BBDD el pc en cuestión y se envía al modelo.

Para la petición de deletePc, únicamente se necesita el id del pc a eliminar, con este dato se llama al método correspondiente del servicio y luego se vuelve a cargar la vista de pcs actualizada.

El uso de spring caché se nota especialmente al poner los estados de la caché, ya que dado que los estados nunca son modificados en la BBDD, no es necesario volver a cargarlos en la caché y siempre se consultan de nuevo desde la caché.