Lógica de negocio con Spring

Ciclo 3: Desarrollo de Software

AÑO 2022

Introducción

El presente documento se ha elaborado teniendo como objetivo la implementación de una lógica de negocio especial. Para esto, se ha utilizado como pretexto la necesidad de entregar información a un reporte. De esta manera el Servicio implementado y el Repositorio no solamente gestionarán las acciones de CRUD sino que tratarán la información que entregan. Para la realización de este tutorial se continuará sobre la base de lo construido en el documento Tutorial de uso JPA y Spring.

Lógica de negocio

La implementación de lógica de negocio, involucrará realizar las siguientes modificaciones:
-Controlador: establecerá la puerta de ingreso a la solicitud de la información para un reporte.
-Servicio: Construirá la respuesta entregada al controlador. Para eso organizará la información que obtiene de la consulta a la base de datos.
-Repositorio: Invocará el nuevo método de CrudRepository
-CrudRepository:Ejecutará la nueva sentencia que se encarga de hacer la búsqueda.

Se creará también una clase en la que transferiremos datos a lo largo de la aplicación.

OBJETIVO
Reportar la cantidad de elementos de la tabla Costumes agrupados por el campo year.

DESARROLLO
Crearemos una clase en la que viajará la información de la dupla Year-Cantidad.

```
public class YearAmount {

    private int year;
    private int amount;

    public YearAmount(int year,int amount){
        this.year=year;
        this.amount=amount;
    }

    public int getYear() {
        return year;
    }

    public void setYear(int year) {
        this.year = year;
    }

    public int getAmount() {
        return amount;
    }

    public void setAmount(int amount) {
        this.amount = amount;
    }
}
```

Ahora, crearemos la consulta en CrudRepository para agrupar. Haremos la consulta con JPQL (Java Persistence Query Language).

La consulta en SQL corresponde a:

```
SELECT  costume.id, count(*) as "total" FROM costume GROUP BY
costume.year ORDER BY total DESC;
```

De este modo la interfaz [CostumeCrudRepository se vería así:

```
public interface CostumeCrudRepository extends CrudRepository<Costume,Integer> {

    @Query("SELECT c.year, COUNT(c.year) from Costume AS c group by c.year order by COUNT(c.year) DESC")
    public List<Object[]> countTotalCostumesByYear();

}
```

Spring se encargará de ejecutar la sentencia en el momento en que se invoque el método llamado countTotalCostumesByYear. El resultado será una lista de un arreglo de objetos. En cada posición del arreglo estarán los campos que solicita el select. En este caso, en la primera posición del arreglo estará YEAR y en la segunda estará el conteo.

Crearemos ahora en la clase CostumeRepository un método que invoque la ejecución de la consulta:

```
public List<Object[]> getTopByYear(){
    return costumeCrudRepository.countTotalCostumesByYear();
}
```

A continuación, haremos que el servicio organice la información para presentarla al controlador.

```
public List<YearAmount> getTopCostumesByYear(){
    List<Object[]> report=costumeRepository.getTopByYear();
    List<YearAmount>res=new ArrayList<>();
    for(int i=0;i<report.size();i++){
        res.add(new YearAmount((int)report.get(i)[0],(int) report.get(i)[1]));
    }
    return res;
}
```

En este proceso, lo que hacemos es almacenar el resultado en report. Luego la respuesta la escribiremos en res. El ciclo corresponde a la conversión de cada dato para crear objetos de tipo YearAmount, los cuales serán el conjunto de respuestas, que representan la dupla Año-Cantidad.

A continuación se modificará el controlador para agregar un llamado GET con su respectiva url, con el fin de poder acceder a esta funcionalidad.

```
//Report!
@GetMapping("/report")
public List<YearAmount> getReport(){
    return costumeService.getTopCostumesByYear();
}
```

De esta manera, el controlador responderá la lista de pareja Año-Cantidad en formato JSON entregada por el servicio.