Ajustando nuestras clases con JsonView

Elementos avanzados en tu API REST con Spring Boot

JsonViews

- Jackson 2 + Spring nos proporcionan un mecanismo para seleccionar qué campos de un objeto serán transformados a JSON.
- Este mecanismo permite tener, para un solo objeto Java diferentes representaciones en JSON.
- Muy cómodo para diferenciar 2 vistas para un objeto
    - *Conjunto de datos para un listado*
    - *Vista de detalle.*

JsonViews

- No es, forzosamente, un mecanismo alternativo al uso de DTO.
- Podemos usarlo tanto con Entidades como con DTO.

Recordemos aquí que los expertos nos recomiendan a tratar de no usar Entidades en las peticiones y respuestas de nuestra API, si bien no existe ninguna obligación de hacerlo.

¿Cómo funciona?

- Definimos una clase que tendrá dentro varias interfaces.
- Serán las diferentes vistas que tendremos para una clase/entidad
- Las interfaces pueden heredar unas de otras

```
public class ProductoViews {

     public interface Dto { }

     public interface DtoConPrecio extends Dto { }

}
```

¿Cómo funciona?

- En nuestro modelo, usamos @JsonView sobre los diferentes atributos
- Definimos, para cada uno, en qué vista o vistas lo vamos a querer obtener.

```
public class ProductoDTO {
        @JsonView(ProductoViews.Dto.class) private String imagen;
        @JsonView(ProductoViews.DtoConPrecio.class) private float
precio;
}
```

¿Cómo funciona?

- Es posible tener atributos que no tengan ningún @JsonView asociado.
- Estos atributos se obtendrán siempre.

¿Cómo funciona?

- Anotamos a nivel de método, con la vista que queremos obtener

```
@JsonView(ProductoViews.DtoConPrecio.class)
@GetMapping(value = "/producto")
public ResponseEntity<?> buscarProductosPorVarios(...) {
```

- También añadimos en *application.properties*

```
spring.jackson.mapper.default-view-inclusion=true
```

¿Cuándo utilizarlo?

- Diferentes métodos en el controlador devuelven diferentes *vistas* de una misma clase (Maestro/Detalle)
- En función del ROL del usuario que hace la petición, queremos devolver más o menos campos.
    - Por ejemplo: Información de los empleados de una empresa
        - ROL USER: información básica
        - ROL ADMIN: información más completa
        - ROL HR: información sobre salario, cotizaciones, ...

Reto

- Trata de jugar con los datos que se obtienen, a diferentes niveles, definiendo varias vistas diferentes.
- En este punto, quizás sería bueno ampliar el modelo de producto, para tener diferentes atributos con los que poder jugar.
- Si quieres tener más datos de ejemplo, puedes utilizar el servicio https://mockaroo.com/ para generar datos de ejemplo (es con el que se han generado los datos actuales).