Perfiles

Elementos avanzados en tu API REST con Spring Boot

Perfiles

- Los perfiles de Spring permiten un mecanismo para para separar determinadas partes en la configuración de un proyecto.
- Podemos hacer que cada una de esas partes esté disponible en un determinado entorno
  - Desarrollo (*dev*)
  - Testing (*test*)
  - Producción (*prod*)
  - ...

@Profile

- Anotación que puede acompañar a
  - @Component (y sus derivados)
  - @Configuration
  - @ConfigurationProperties
- A través de ella indicamos el perfil o perfiles en los cuales tendremos a nuestra disposición dicho componente

@Profile

```
@Configuration
@Profile("production")
public class ProductionConfiguration {
        @Bean
        public Datasource datasource() {
    // ...
        }
}
```

Perfiles y *properties*

- La propiedad *spring.profiles.active* nos permite indicar el perfil (o perfiles) activos
- Se puede especificar de diferentes formas.
- La más sencilla durante desarrollo es en *application.properties*

```
spring.profiles.active=dev,mysqldb
```

Perfiles y *properties*

- También podemos especificarlo a través del *pom.xml*

```
<plugins>
    <plugin>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-maven-plugin</artifactId>
        <configuration>
            <profiles>
                <profile>dev</profile>
            </profiles>
        </configuration>
    </plugin>
    ...
</plugins>
```

Perfiles y *properties*

- También a través de un parámetro de la JVM
  - *-Dspring.profiles.active=dev*
- O incluso de una variable de entorno (la forma depende del sistema operativo)
  - *SPRING_PROFILES_ACTIVE=dev*

Propiedades específicas de un perfil

- Muy útil para configurar determinadas propiedades de diferente forma según el perfil activo.
- Dejaremos las propiedades comunes en *application.properties*.
- Creamos un nuevo fichero por perfil. El nombre será *application-{profile}.properties*. Por ejemplo:
  - *application-dev.properties*
  - *application-prod.properties*

Reto

- Realizar esta tarea con una copia del proyecto ManyToMany para configurar más de un perfil.
- Como segundo reto, puedes proponerte utilizar dos bases de datos Postgresql como entorno local y de producción
    - Para ello, necesitas crear dos contenedores con docker
    - Los pasos para crear uno lo vimos en la lección anterior.
    - El segundo contenedor deberá tener un nombre y un puerto local diferente.