

EJERCICIO 1 – CICLO DE VIDA DE UN BEAN

Objetivo: Implementar los métodos de ciclo de vida de un bean utilizando las anotaciones `@PostConstruct` y `@PreDestroy`.

1. Crea una clase `DatabaseConnection` que simule una conexión a una base de datos. Utiliza `@PostConstruct` para que, al inicializar el bean, se conecte a la base de datos.
2. Utiliza `@PreDestroy` para que, al destruir el bean, cierre la conexión a la base de datos.
3. Configura un contexto de Spring para crear el bean y ver cómo se ejecutan los métodos de ciclo de vida.

Usa un método `connect()` que imprima "Conexión a la base de datos establecida" y un método `disconnect()` que imprima "Conexión a la base de datos cerrada".

EJERCICIO 2: USAR LA ANOTACIÓN `@RESOURCE` PARA LA INYECCIÓN DE DEPENDENCIAS

Objetivo: Inyectar dependencias usando `@Resource` en lugar de `@Autowired`.

1. Crea una interfaz `INotificacionServicio` y dos implementaciones: `EmailService` y `SMSService`.
2. Crea una clase `NotificationController` que tenga un campo `INotificacionServicio`.
3. Usa `@Resource` para inyectar una de las implementaciones de `NotificationService`.
4. Configura las clases en Spring para que se inyecten automáticamente y verifica cuál implementación se inyecta.

EJERCICIO 3: USAR LA ANOTACIÓN `@VALUE` PARA INYECTAR VALORES DE PROPIEDADES

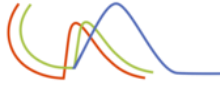
Objetivo: Inyectar valores desde un archivo de propiedades usando la anotación `@Value`. Los valores son nombre de la aplicación, versión,

EJERCICIO 4: CONFIGURAR EL ÁMBITO SINGLETON

Objetivo: Comprobar el comportamiento del ámbito singleton en Spring.

- Crea una clase `UserService` con un método `getServiceId()` que devuelva el hash del objeto.
- Declara la clase como un bean singleton (que es el valor por defecto en Spring).
- En la clase principal, inyecta el mismo bean en dos variables distintas y compara si son la misma instancia.
- Imprime el `hashCode` del objeto en ambas inyecciones y verifica que sea el mismo.

EJERCICIO 5: CONFIGURAR EL ÁMBITO PROTOTYPE



Objetivo: Configurar un bean con ámbito prototype y comprobar que Spring crea nuevas instancias en cada inyección.

1. Crea una clase Task con un método `getTaskId()` que devuelva el hash del objeto.
2. Declara el bean como de ámbito prototype usando la anotación `@Scope`.
3. Inyecta el bean en dos variables diferentes dentro de una clase y comprueba si son instancias distintas.
4. Imprime el `hashCode` del objeto en ambas inyecciones para verificar que son diferentes instancias.