



EJERCICIO 1: CONFIGURACIÓN DE COMPONENT SCANNING

Configurar Spring para escanear automáticamente un paquete en busca de componentes anotados con `@Component`, `@Service`, `@Repository`, etc., utilizando la anotación `@ComponentScan`.

Instrucciones:

1. Crear dos clases en diferentes paquetes anotadas con `@Component`.
2. Configurar Spring para que escanee automáticamente esos paquetes con la anotación `@ComponentScan`.
3. Recuperar y utilizar los beans escaneados en una aplicación Spring.

EJERCICIO 2: USAR LA ANOTACIÓN @VALUE

Objetivo: Inyectar valores desde un archivo de propiedades usando `@Value`, pero en esta ocasión calcular valores en base a las propiedades inyectadas.

1. Crea un archivo `application.properties` con los valores de `app.author` y `app.year`.
2. Crea una clase `AppMetadata` con campos para el nombre del autor y el año de creación de la aplicación.
3. Usa `@Value` para inyectar estos valores y crea un método que combine los valores para generar un string como "Aplicación creada por {autor} en {año}".
4. Muestra este mensaje en la clase principal.

EJERCICIO 3 CAMBIAR EL NOMBRE DEL ARCHIVO APPLICATION.PROPERTIES EN UN PROYECTO DE SPRING BOOT, AUNQUE NO ES LO MÁS COMÚN.

Usa `spring.config.name`

EJERCICIO 4: CICLO DE VIDA DE UN BEAN

Crea una clase `ExternalService` que simule la conexión a un servicio externo.

Usa `@PostConstruct` para ejecutar el método `initialize()` que imprima "Conexión a servicio externo establecida".

Usa `@PreDestroy` para ejecutar el método `shutdown()` que imprima "Conexión a servicio externo cerrada".

Configura el contexto de Spring para crear el bean y ver cómo se ejecutan los métodos de ciclo de vida.