Universidad Nacional Autónoma de México

Dirección General de Cómputo de Tecnologías de Información y Comunicación

Diplomado

Desarrollo de Sistemas con Tecnología Java

Ejercicio Once

“Inicializar y Destruir un Bean”

Mtro. ISC. Miguel Ángel Sánchez Hernández

Tabla de contenido

1. Copiar un proyecto en IntelliJ IDEA

Para copiar un proyecto, hacemos lo siguiente:

1. Señalamos el proyecto spring-core-scope
2. Apretamos Ctrl + C
3. Luego Ctrl + V
4. New Name: spring-core-initdestroy
5. En el archivo pom.xml cambiar las siguientes líneas
    - <artifactId>spring-core-scope</artifactId> por lo siguiente
        - <artifactId>spring-core-initdestroy</artifactId>
    - Cambiar las etiquetas <name> con
        - <name> spring-core-initdestroy </name>
    - Cambiar la etiqueta <description> con
        - <description> Ejemplo de inicializar y destruir bean con Spring </description>
6. Actualizar maven

2. Crear la clase Servicio

Crear la clase Servicio como sigue:

```
package dgtic.core.servicio;
import dgtic.core.modelo.Empleado;
public class Servicio {
		private static Servicio servicio=new Servicio();
		private Empleado empleado;
		private Servicio() {
		}
		public static Servicio getInstance() {
			return servicio;
		}
		public Empleado getEmpleado() {
			return empleado;
		}
		public void setEmpleado(Empleado empleado) {
			this.empleado = empleado;
		}

		public void iniciar() {
			System.out.println("Iniciando Servicio...");
		}
		public void destruir() {
			System.out.println("Liberando recursos de Servicio...");
		}
}
```

3. Crear la clase Empleado

Crear la clase Empleado como sigue:

```
package dgtic.core.modelo;
import org.springframework.beans.factory.InitializingBean;
public class Empleado implements InitializingBean{
		private String nombre;
		private Integer edad;
		private Actividades actividad;
		public Empleado() {
		}

		public Empleado(Actividades actividad) {
			super();
			this.actividad = actividad;
		}

		public Empleado(String nombre) {
			super();
			this.nombre = nombre;
		}

		public String getNombre() {
			return nombre;
		}
		public void setNombre(String nombre) {
			this.nombre = nombre;
		}
		public Integer getEdad() {
			return edad;
		}
		public void setEdad(Integer edad) {
			this.edad = edad;
		}

		public Actividades getActividad() {
			return actividad;
		}

		public void setActividad(Actividades actividad) {
			this.actividad = actividad;
		}
		@Override
		public String toString() {
			return "Empleado [nombre=" + nombre + ", edad=" + edad + "]";
		}
		public void limpiar(){
			System.out.println("Liberando recursos de Empleado-----");
		}
		@Override
		public void afterPropertiesSet() throws Exception {
			System.out.println("Iniciando Empleado-----");
		}
}
```

4. Archivo de configuración bean-configuration.xml (versión 1)

Modificamos el archivo bean-configuration.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
	xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd">
	<!-- bean definitions here -->
	<bean id="empleado" class="dgtic.core.modelo.Empleado"
                                                scope="prototype"
                                                destroy-method="limpiar"> <!-- cambiar scope -->
		<property name="nombre" value="Ramon" />
		<property name="edad" value="27" />
	</bean>
</beans>
```

5. Archivo de configuración bean-services.xml

Crear el archivo bean-services.xml, en el paquete dgtic.core.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
	xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd">
	<!-- bean definitions here -->
	<bean id="servicio" class="dgtic.core.servicio.Servicio"
                                factory-method="getInstance"
                                init-method="iniciar"
                                destroy-method="destruir">
		<property name="empleado" ref="empleado" />
	</bean>
</beans>
```

6. Clase Inicio (versión 1)

Modificamos la clase Inicio como sigue:

```
package dgtic.core;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import dgtic.core.modelo.Empleado;
import dgtic.core.servicio.Servicio;
public class Inicio {
	public static void main(String[] args) {
		ApplicationContext contexto = new ClassPathXmlApplicationContext(
			new String[] { "/src/main/java/dgtic/core/xml/bean-configuration.xml", "/dgtic/core/xml/bean-
services.xml" });
		Empleado emp = (Empleado) contexto.getBean("empleado");
		System.out.println("+++++");
		Servicio serv = (Servicio) contexto.getBean("servicio");
		((ClassPathXmlApplicationContext) contexto).close();
	}
}
```

Correar la aplicación. Podemos ver que el bean “servicio”, llama sus métodos para inicializar y terminar, esto se debe a que ocupa el patrón singleton, pero el bean “empleado” no hace eso, y es debido a que Spring no gestiona el ciclo de vida completo de un bean prototype.

“El contenedor instancia, configura, decora y ensambla un objeto prototype, se lo entrega al cliente y luego no tiene más conocimiento de esa instancia prototype. Esto significa que, si bien se llamará a los métodos de devolución de llamada del ciclo de vida de inicialización en todos los objetos independientemente del alcance, en el caso de los prototypes, no se llamará a ninguna devolución de llamada del ciclo de vida de destrucción configurada. Es responsabilidad del código del cliente limpiar los objetos del ámbito del prototype y liberar los recursos costosos que retienen los beans prototipo (Documentación de Spring).”

7. Clase Inicio (versión 2)

Modificamos la clase Inicio como sigue:

```
package dgtic.core;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import dgtic.core.modelo.Empleado;
import dgtic.core.servicio.Servicio;
public class Inicio {
		public static void main(String[] args) {
			ApplicationContext contexto = new ClassPathXmlApplicationContext(
					new String[] { "/src/main/java/dgtic/core/xml/bean-configuration.xml", "/dgtic/core/xml/bean-
services.xml" });
			Empleado emp = (Empleado) contexto.getBean("empleado");
			System.out.println("+++++");
			Servicio serv = (Servicio) contexto.getBean("servicio");
			emp.limpiar();
			((ClassPathXmlApplicationContext) contexto).close();
		}
}
```

Correr la aplicación. Nosotros hacemos la llamada explicita para liberar los recursos.

8. Clase Inicio (versión 3)

Modificamos la clase Inicio como sigue:

```
package dgtic.core;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import dgtic.core.modelo.Empleado;
import dgtic.core.servicio.Servicio;
public class Inicio {
		public static void main(String[] args) {
			ApplicationContext contexto = new ClassPathXmlApplicationContext(
					new String[] { "/src/main/java/dgtic/core/xml/bean-configuration.xml", "/dgtic/core/xml/bean-
services.xml" });
			Empleado emp = (Empleado) contexto.getBean("empleado");
			System.out.println("+++++");
			Servicio serv = (Servicio) contexto.getBean("servicio");
				((ClassPathXmlApplicationContext) contexto).close();
	}
}
```

9. Archivo de configuración bean-configuration.xml (versión 2)

Modificamos el archivo bean-configuration.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
	xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd">
	<!-- bean definitions here -->
	<bean id="empleado" class="dgtic.core.modelo.Empleado"
			scope="singleton">
			<property name="nombre" value="Ramon" />
			<property name="edad" value="27" />
	</bean>
</bean>
```

Correr la aplicación. Cada bean Empleado y Servicio se llama sus métodos de inicialización y termino.