In Spring, **Bean Scopes** define the lifecycle and visibility of beans within the Spring container. There are several scopes available in Spring, each of which controls how and when a bean is created, shared, and destroyed. The most common scopes are:

1. **Singleton** (default)
2. **Prototype**
3. **Request**
4. **Session**
5. **Global Session** (used in a portlet context)

Let's go through examples for the most common bean scopes, which are **Singleton** and **Prototype**.

**1. Singleton Scope (Default)**

The default scope in Spring is singleton, which means the Spring container creates only one instance of the bean, and that same instance is used throughout the application.

**Example: Singleton Scope**

**Java Configuration:**

java

Copy

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

@Configuration

public class AppConfig {

@Bean

public MyBean myBean() {

return new MyBean();

}

}

**MyBean Class:**

java

Copy

public class MyBean {

public MyBean() {

System.out.println("MyBean instantiated!");

}

public void doSomething() {

System.out.println("Doing something...");

}

}

**Main Application:**

java

Copy

import org.springframework.context.annotation.AnnotationConfigApplicationContext;

public class Main {

public static void main(String[] args) {

AnnotationConfigApplicationContext context = new AnnotationConfigApplicationContext(AppConfig.class);

MyBean bean1 = context.getBean(MyBean.class);

MyBean bean2 = context.getBean(MyBean.class);

// Check if they are the same instance

System.out.println(bean1 == bean2); // Will print: true

context.close();

}

}

**Output:**

arduino

Copy

MyBean instantiated!

true

As expected, the same instance of MyBean is shared in the singleton scope.

**2. Prototype Scope**

With the prototype scope, Spring creates a new instance of the bean every time it is requested. This means each time you call getBean(), a new instance is created.

**Example: Prototype Scope**

**Java Configuration:**

java

Copy

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

@Configuration

public class AppConfig {

@Bean

public MyBean myBean() {

return new MyBean();

}

}

**MyBean Class:**

java

Copy

public class MyBean {

public MyBean() {

System.out.println("MyBean instantiated!");

}

public void doSomething() {

System.out.println("Doing something...");

}

}

**Main Application:**

java

Copy

import org.springframework.context.annotation.AnnotationConfigApplicationContext;

public class Main {

public static void main(String[] args) {

AnnotationConfigApplicationContext context = new AnnotationConfigApplicationContext(AppConfig.class);

// Requesting beans with prototype scope

MyBean bean1 = context.getBean(MyBean.class);

MyBean bean2 = context.getBean(MyBean.class);

// Check if they are different instances

System.out.println(bean1 == bean2); // Will print: false

context.close();

}

}

**Output:**

arduino

Copy

MyBean instantiated!

MyBean instantiated!

false

In this case, two separate instances of MyBean are created because of the prototype scope.

**3. Request Scope (Web Applications Only)**

In a web application, **request scope** means a new bean instance is created for each HTTP request.

**Example: Request Scope**

**Java Configuration:**

java

Copy

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.web.context.request.RequestContextListener;

@Configuration

public class AppConfig {

@Bean

public RequestContextListener requestContextListener() {

return new RequestContextListener();

}

@Bean

@Scope("request")

public RequestBean requestBean() {

return new RequestBean();

}

}

**RequestBean Class:**

java

Copy

public class RequestBean {

public RequestBean() {

System.out.println("RequestBean instantiated!");

}

public void handleRequest() {

System.out.println("Handling request...");

}

}

**Main Application (Spring Boot setup for demonstration):**

java

Copy

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class MyApp {

public static void main(String[] args) {

SpringApplication.run(MyApp.class, args);

}

}

Here, every HTTP request will create a new instance of RequestBean. However, this scope is mostly used in Spring Web applications.

**4. Session Scope (Web Applications Only)**

In a web application, **session scope** ensures a bean is scoped to a single HTTP session. A new instance will be created for each session.

**Example: Session Scope**

java

Copy

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.web.context.annotation.SessionScope;

@Configuration

public class AppConfig {

@Bean

@SessionScope

public SessionBean sessionBean() {

return new SessionBean();

}

}

**SessionBean Class:**

java

Copy

public class SessionBean {

public SessionBean() {

System.out.println("SessionBean instantiated!");

}

public void handleSession() {

System.out.println("Handling session...");

}

}

This scope is only relevant in web applications, where each user's session has its own instance of the bean.

**5. Global Session Scope (Web Applications Only)**

This scope is used only in a **Portlet** environment. It ensures a single bean is shared across the global session.

**Example: Global Session Scope**

java

Copy

@Bean

@Scope("globalSession")

public GlobalSessionBean globalSessionBean() {

return new GlobalSessionBean();

}

This scope is very specific and used in portlet-based applications (not typical for standard Spring applications).

**Conclusion**

* **Singleton Scope**: One instance per Spring container (default).
* **Prototype Scope**: A new instance every time the bean is requested.
* **Request Scope**: A new bean instance for each HTTP request (only in web applications).
* **Session Scope**: A new bean instance for each HTTP session (only in web applications).
* **Global Session Scope**: Used in portlet applications.

Each of these scopes can be set using the @Scope annotation in a Spring configuration class or defined in XML configuration files.