JAVA SPRING FRAMEWORK

Lab Guides

Document Code	**25e-BM/HR/HDCV/FSOFT**
Version	**1.0**
Effective Date	**01/05/2022**

Hanoi, 08/2024

RECORD OF CHANGES

No	Effective Date	Change Description	Reason	Reviewer	Approver
1	06/08/2024	Create a new Lab	Create new		VinhNV

Contents

CODE:	JSFW_Lab_02_Opt3
TYPE:	SHORT
LOC:	200
DURATION:	60 MINUTES

Java Spring Framework Introduction

Objectives:

- Understand how to use session scope with Spring beans.
- Learn to configure and use beans with prototype scope in a real-life scenario.

Lab Specifications:

In a Shopping Cart Application, the ShoppingCart bean is session-scoped since each user's session should maintain its own shopping cart.

Problem Description:

- Trainees must write scripts to test the methods they have developed.

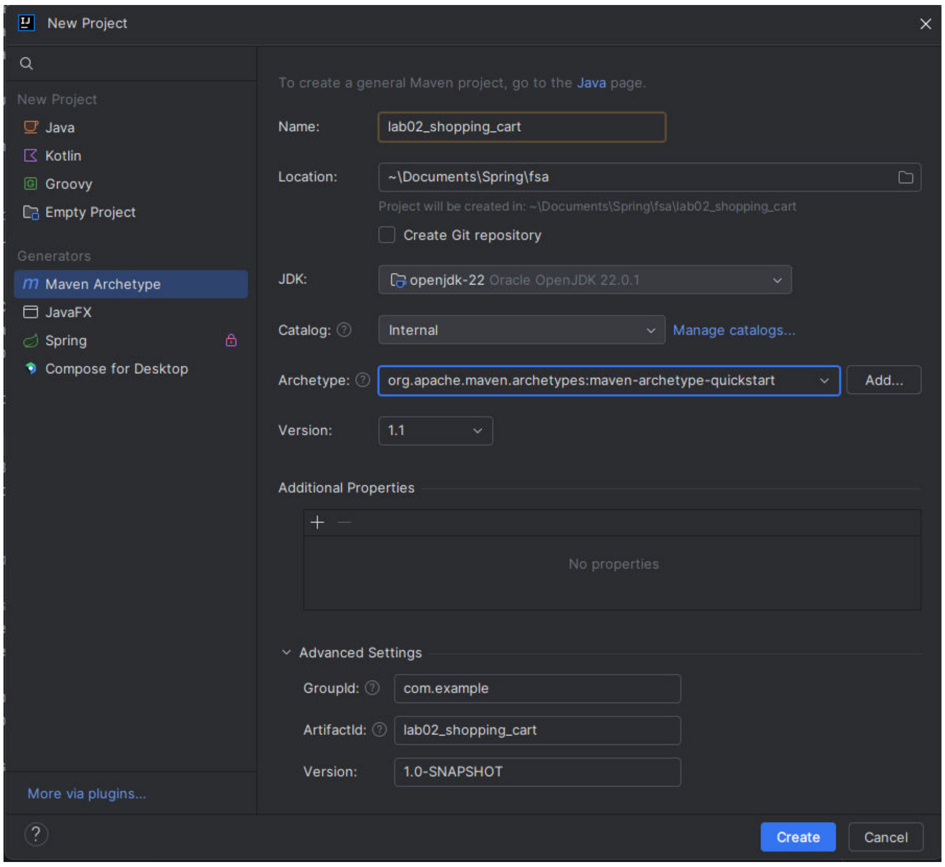
Prerequisites:

- Using Java SDK version 8.0 at least.
- Using Maven.
- Using Spring Framework 5.0 or higher version.

Guidelines:

Step 1: Extend the previous project to include dependency injection:

- Open IntelliJ IDEA.
- Click on File -> New -> Project....
- Select Maven from the project types.
- Click Next and set the project name to **lab02_shopping_cart**.
- Set the groupId to com.example and artifactId to **lab02_ shopping_cart**.
- Click Create.


New Project

New Project
Java
Kotlin
Groovy
Empty Project

Generators
Maven Archetype
JavaFX
Spring
Compose for Desktop

More via plugins...

To create a general Maven project, go to the Java page.

Name: lab02_shopping_cart

Location: ~\Documents\Spring\fsa

Project will be created in: ~\Documents\Spring\fsa\lab02_shopping_cart

Create Git repository

JDK: openjdk-22 Oracle OpenJDK 22.0.1

Catalog: Internal

Archetype: org.apache.maven.archetypes:maven-archetype-quickstart

Version: 1.1

Additional Properties

No properties

Advanced Settings

GroupId: com.example

ArtifactId: lab02_shopping_cart

Version: 1.0-SNAPSHOT

Step 2: Add dependencies and configuration into pom.xml file:

```
<parent>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-parent</artifactId>
  <version>2.7.5</version>
  <relativePath/>
</parent>
```

Add the Spring Core dependency to your pom.xml file.

```
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-context</artifactId>
  <version>5.3.20</version>
</dependency>
```

Step 3: Create entity classes:

Create UniversityService class:

```
package com.example;

import java.util.ArrayList;
import java.util.List;

public class ShoppingCart {
    private List<String> items = new ArrayList<>();

    public void addItem(String item) {
        items.add(item);
    }

    @Override
    public String toString() {
        return "ShoppingCart{" +
                "items=" + items +
                '}';
    }
}
```

Step 4: Configure Beans with Session Scope.

Create a configuration class AppConfig

```
package com.example;

import org.springframework.beans.factory.config.CustomScopeConfigurer;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.context.annotation.Scope;
import org.springframework.context.support.SimpleThreadScope;

@Configuration
public class AppConfig {

    @Bean
    @Scope("session")
    public ShoppingCart shoppingCart() {
        System.out.println("A new ShoppingCart instance created");
        return new ShoppingCart();
    }

    @Bean
    public static CustomScopeConfigurer customScopeConfigurer() {
        CustomScopeConfigurer configurer = new CustomScopeConfigurer();
        configurer.addScope("session", new SimpleThreadScope());
        return configurer;
    }
}
```

Note: To properly test the session scope in a non-web environment, we need to manually register a custom scope. In a web application, the session scope is typically managed by the web container, but for standalone applications, you need to configure it manually.

Step 5: Create a Main Class to Test the Singleton Scope:

Create a **MainApp** class:

```
package com.example;

import org.springframework.context.ApplicationContext;
import
org.springframework.context.annotation.AnnotationConfigApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import org.springframework.context.support.AbstractApplicationContext;

public class MainApp {
    public static void main(String[] args) {
        ApplicationContext context = new
AnnotationConfigApplicationContext(AppConfig.class);
        //ApplicationContext context = new
ClassPathXmlApplicationContext("beans.xml");
```

```
        ShoppingCart cart1 = (ShoppingCart) context.getBean("shoppingCart");
        ShoppingCart cart2 = (ShoppingCart) context.getBean("shoppingCart");

        cart1.addItem("Item1");
        cart2.addItem("Item2");

        System.out.println(cart1);
        System.out.println(cart2);

        ((AbstractApplicationContext) context).close();
    }
}
```

Step 6: Run the Application:

- Run the MainApp.java class.
- Verify that it prints:

```
A new ShoppingCart instance created

ShoppingCart{items=[Item1, Item2]}

ShoppingCart{items=[Item1, Item2]}
```

Step 7: Write a JUnit Test Case:

1. Update porm.xml

```
<dependency>
  <groupId>org.junit.jupiter</groupId>
  <artifactId>junit-jupiter</artifactId>
  <version>RELEASE</version>
  <scope>compile</scope>
</dependency>
```

2. Create a test class **ShoppingCartSessionScopeTest**.java.

```
package com.example;

import org.junit.jupiter.api.Test;
import org.springframework.beans.factory.config.CustomScopeConfigurer;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import org.springframework.context.support.SimpleThreadScope;

import static org.junit.jupiter.api.Assertions.assertNotSame;

public class ShoppingCartSessionScopeTest {

    @Test
    public void testSessionScope() {
        ClassPathXmlApplicationContext context = new
ClassPathXmlApplicationContext("beans.xml");

        // Register session scope with context
        CustomScopeConfigurer configurer = new CustomScopeConfigurer();
        configurer.addScope("session", new SimpleThreadScope());
        configurer.postProcessBeanFactory(context.getBeanFactory());

        ShoppingCart cart1 = (ShoppingCart) context.getBean("shoppingCart");

        // Simulate end of session by manually clearing the scoped beans
        context.getBeanFactory().destroyScopedBean("shoppingCart");
        ShoppingCart cart2 = (ShoppingCart) context.getBean("shoppingCart");

        // Since it's a new session, cart1 and cart2 should not be the same
instance
        assertNotSame(cart1, cart2, "The two ShoppingCart beans should be
different instances in different sessions");

        // Add items and check state
        cart1.addItem("Item1");
        cart2.addItem("Item2");

        System.out.println(cart1);
        System.out.println(cart2);

        context.close();
    }
}
```

3. Run the test and verify it passes.

This exercise will help you understand how to use prototype scope in Spring with real-life scenarios such as managing employees and customers. The key takeaway is that each time a prototype-scoped bean is requested, a new instance is created, allowing for independent management of each entity.

----oOo-----

THE END