**Exercise-1:Configuring a Basic Spring Application**

<!-- pom.xml: Add Spring Core Dependency -->

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.34</version>

</dependency>

</dependencies>

<!-- applicationContext.xml: Define Spring Beans -->

<?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">

<!-- Repository Bean -->

<bean id="bookRepository" class="com.library.repository.BookRepository" />

<!-- Service Bean with dependency injection -->

<bean id="bookService" class="com.library.service.BookService">

<property name="bookRepository" ref="bookRepository"/>

</bean>

</beans>

// BookRepository.java: Handles book data operations

package com.library.repository;

public class BookRepository {

public void saveBook(String bookName) {

System.out.println("Book saved: " + bookName);

}

}

// BookService.java: Service layer to use BookRepository

package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

private BookRepository bookRepository;

// Setter for dependency injection

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

// Business logic to add a book

public void addBook(String bookName) {

bookRepository.saveBook(bookName);

}

}

// LibraryApp.java: Main class to load Spring context and run the application

package com.library.main;

import com.library.service.BookService;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class LibraryApp {

public static void main(String[] args) {

// Load Spring context from XML file

ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");

// Get BookService bean and use it

BookService bookService = context.getBean("bookService", BookService.class);

bookService.addBook("The Great Gatsby");

// Close the context

((ClassPathXmlApplicationContext) context).close();

}

}

**Exercise-2: Implementing Dependency Injection**

<!-- pom.xml: Add Spring Core Dependency -->

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.34</version>

</dependency>

</dependencies>

<!-- applicationContext.xml: Define Spring Beans and wire dependencies -->

<?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">

<!-- Repository Bean -->

<bean id="bookRepository" class="com.library.repository.BookRepository" />

<!-- Service Bean with dependency injection -->

<bean id="bookService" class="com.library.service.BookService">

<property name="bookRepository" ref="bookRepository"/>

</bean>

</beans>

// BookRepository.java: Handles book data operations

package com.library.repository;

public class BookRepository {

public void saveBook(String bookName) {

System.out.println("Book saved: " + bookName);

}

}

// BookService.java: Service layer to use BookRepository

package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

private BookRepository bookRepository;

// Setter for dependency injection (required for DI in XML)

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

// Business logic to add a book

public void addBook(String bookName) {

bookRepository.saveBook(bookName);

}

}

// LibraryApp.java: Main class to load Spring context and run the application

package com.library.main;

import com.library.service.BookService;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class LibraryApp {

public static void main(String[] args) {

// Load Spring context from XML file

ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");

// Get BookService bean and use it

BookService bookService = context.getBean("bookService", BookService.class);

bookService.addBook("The Great Gatsby");

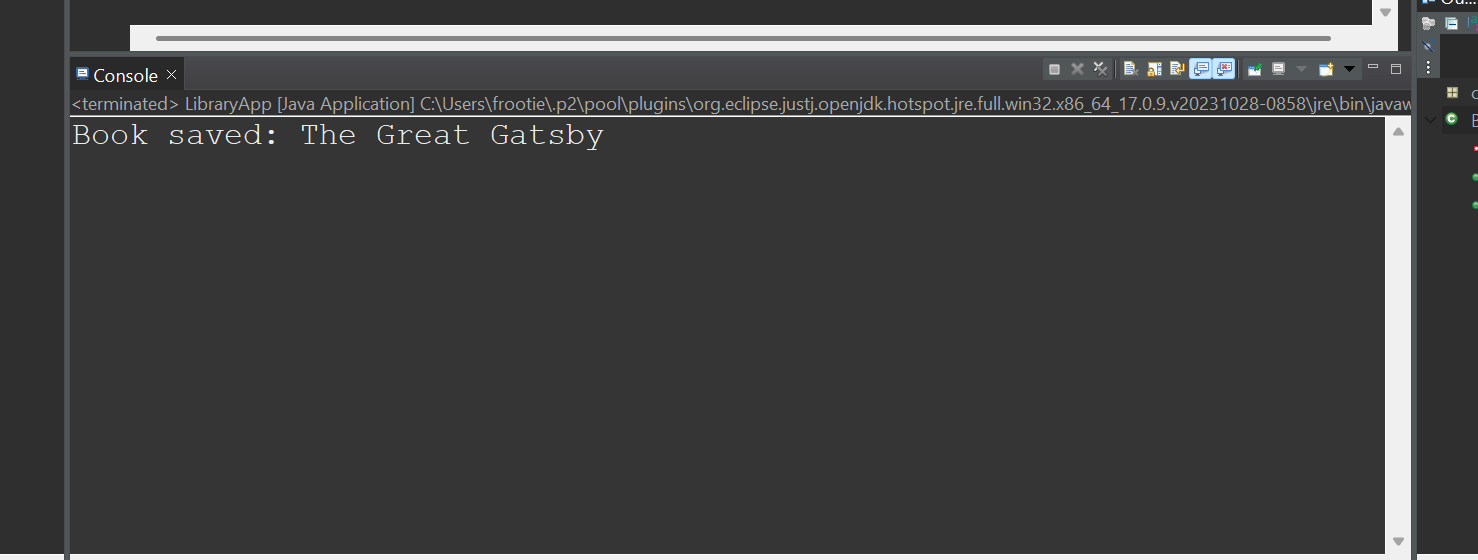
// Close the context

((ClassPathXmlApplicationContext) context).close();

}

}

**OUTPUT:**

****

**Exercise 4: Creating and Configuring a Maven Project**

<!-- Step 1: Create a New Maven Project -->

<!-- Step 2: Add Spring Dependencies to pom.xml -->

<dependencies>

<!-- Spring Context for core container and DI -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.34</version>

</dependency>

<!-- Spring AOP for aspect-oriented programming -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aop</artifactId>

<version>5.3.34</version>

</dependency>

<!-- Spring WebMVC for building web applications -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>5.3.34</version>

</dependency>

</dependencies>

<!-- Step 3: Configure Maven Compiler Plugin -->

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.10.1</version>

<configuration>

<source>1.8</source>

<target>1.8</target>

</configuration>

</plugin>

</plugins>

</build>