**WEEK 3**

**Exercise 1:**

**Configure Basic Spring Application**

**Pom.xml**

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.library</groupId>

<artifactId>LibraryManagement</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<!-- Spring Core -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.32</version>

</dependency>

</dependencies>

</project>

**APPLICATIONCONTEXT.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 id="bookRepository" class="com.library.repository.BookRepository" />

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

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

</bean>

</beans>

**BOOKREPOSITORY.java**

package com.library.repository;

public class BookRepository {

public void saveBook(String bookName) {

System.*out*.println("Book '" + bookName + "' saved to repository.");

}

}

**BOOKSERVICE.JAVA**

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;

}

public void addBook(String bookName) {

System.*out*.println("BookService: Adding book...");

bookRepository.saveBook(bookName);

}

}

**MAINAPP.JAVA**

package com.library;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.library.service.BookService;

public class MainApp {

public static void main(String[] args) {

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

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

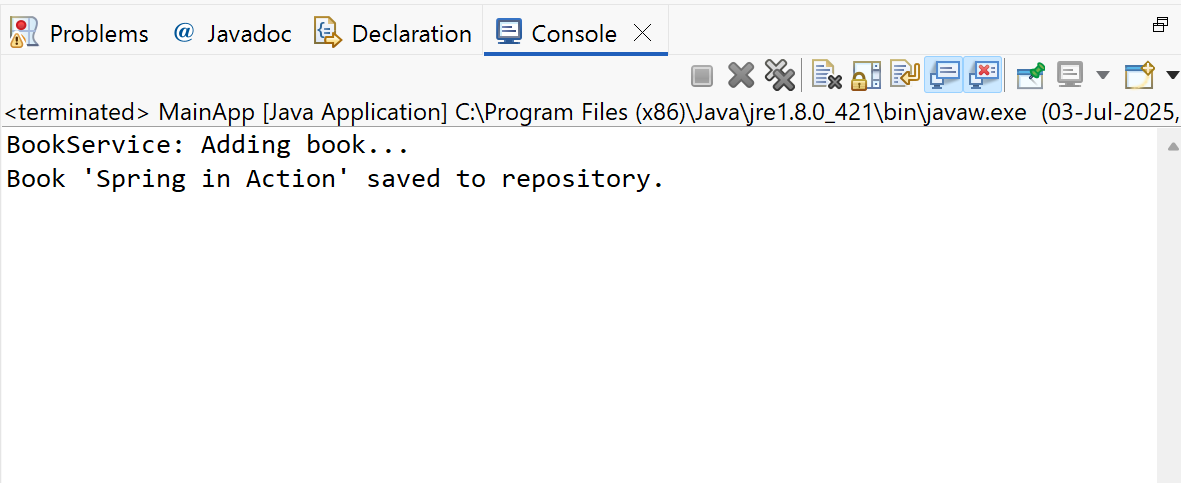
bookService.addBook("Spring in Action");

((ClassPathXmlApplicationContext) context).close();

}

}

**OUTPUT**

****

**EXERCISE 2**

**Implementing Dependency Injection**

**APPLICATIONCONTEXT.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

https://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>

**BOOKSERVICE.JAVA**

package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

private BookRepository bookRepository;

// Required for setter injection

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

public void addBook(String bookName) {

System.*out*.println("BookService: Adding book...");

bookRepository.saveBook(bookName);

}

}

**MAINAPP.JAVA**

package com.library;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.library.service.BookService;

public class MainApp {

public static void main(String[] args) {

// Load Spring context

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

// Get BookService bean (with BookRepository injected)

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

// Test method

bookService.addBook("Effective Java");

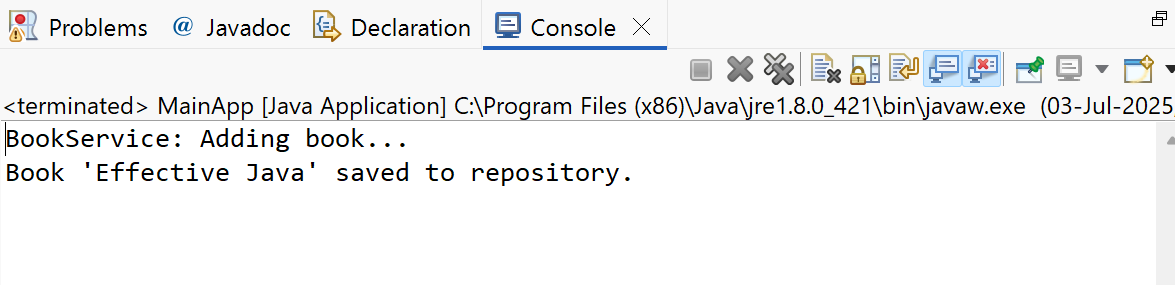
// Close the context

((ClassPathXmlApplicationContext) context).close();

}

}

**OUTPUT**

****

**Exercise 4**

**Creating and Configuring a Maven Project**

**POM.XML**

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.library</groupId>

<artifactId>LibraryManagemenT2</artifactId>

<version>0.0.1-SNAPSHOT</version>

<properties>

<java.version>1.8</java.version>

</properties>

<dependencies>

<!-- Spring Context -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.34</version> <!-- or latest 5.x version -->

</dependency>

<!-- Spring AOP -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aop</artifactId>

<version>5.3.34</version>

</dependency>

<!-- Spring Web MVC -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>5.3.34</version>

</dependency>

<!-- Optional: Servlet API (required for Spring MVC apps) -->

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>javax.servlet-api</artifactId>

<version>4.0.1</version>

<scope>provided</scope>

</dependency>

</dependencies>

<build>

<plugins>

<!-- Maven Compiler Plugin -->

<plugin>

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

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

<version>3.8.1</version>

<configuration>

<source>${java.version}</source>

<target>${java.version}</target>

</configuration>

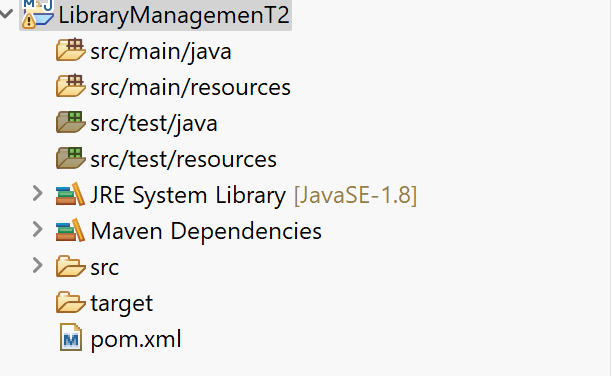
</plugin>

</plugins>

</build>

</project>

**MAVEN PLUGIN JAVA 1.8**

****

**Exercise 5**

**Configuring the Spring IoC Container**

**APPLICATIONCONTEXT.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

https://www.springframework.org/schema/beans/spring-beans.xsd">

<!-- BookRepository Bean -->

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

<!-- BookService Bean with DI -->

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

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

</bean>

</beans>

**BOOKSERVICE.JAVA**

package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

private BookRepository bookRepository;

// Setter method for dependency injection

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

public void addBook(String title) {

System.*out*.println("BookService: Processing book...");

bookRepository.saveBook(title);

}

}

**BOOKREPOSITORY.JAVA**

package com.library.repository;

public class BookRepository {

public void saveBook(String title) {

System.*out*.println("BookRepository: Book '" + title + "' saved.");

}

}

**MAINAPP.JAVA**

package com.library.mainapp;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.library.service.BookService;

public class MainApp {

public static void main(String[] args) {

// Load Spring configuration

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

// Get the BookService bean

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

// Use the service

bookService.addBook("Clean Code");

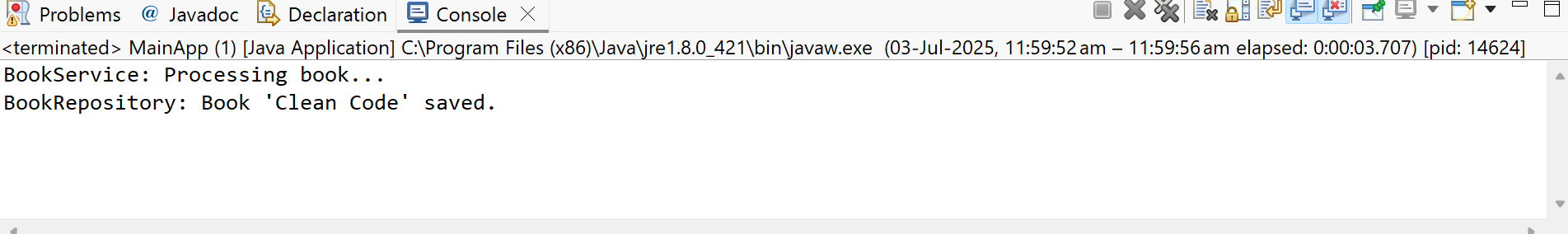
// Close context

((ClassPathXmlApplicationContext) context).close();

}

}

**OUTPUT**

****

**Exercise 7: Implementing Constructor and Setter Injection**

**AAPLICATIONCONTEXT.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

https://www.springframework.org/schema/beans/spring-beans.xsd">

<!-- BookRepository Bean -->

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

<!-- BookService Bean with constructor and setter injection -->

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

<!-- Constructor Injection -->

<constructor-arg value="LibraryServiceV1" />

<!-- Setter Injection -->

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

</bean>

</beans>

**BOOKSERVICE.JAVA**

package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

private BookRepository bookRepository;

private String serviceName; // to be injected via constructor

// Constructor for constructor injection

public BookService(String serviceName) {

this.serviceName = serviceName;

System.*out*.println("Constructor Injection: Service name = " + serviceName);

}

// Setter for setter injection

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

public void addBook(String bookName) {

System.*out*.println("[" + serviceName + "] BookService: Adding book...");

bookRepository.saveBook(bookName);

}

}

**BOOKREPOSITORY.JAVA**

package com.library.repository;

public class BookRepository {

public void saveBook(String bookName) {

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

}

}

**MAINAPP.JAVA**

package com.library.mainapp;

import com.library.service.BookService;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class MainApp {

public static void main(String[] args) {

// Load Spring context

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

// Get bean

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

// Use the service

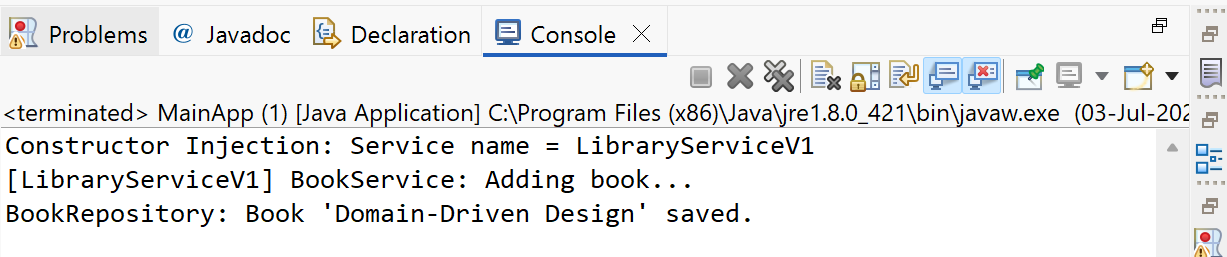
bookService.addBook("Domain-Driven Design");

((ClassPathXmlApplicationContext) context).close();

}

}

**OUTPUT**

****