**Spring Core & Maven**

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

## **Scenario:** Your company is developing a web application for managing a library. You need to use the Spring Framework to handle the backend operations.

### ***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>1.0-SNAPSHOT</version>   <properties>  <java.version>21</java.version>  </properties>   <dependencies>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-context</artifactId>  <version>6.1.6</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>*** |
| --- |

### ***BookService.java:***

| ***package com.library.repository;***  ***public class BookRepository {  public void saveBook(String bookName) {  System.out.println("Book '" + bookName + "' saved.");  } }*** |
| --- |

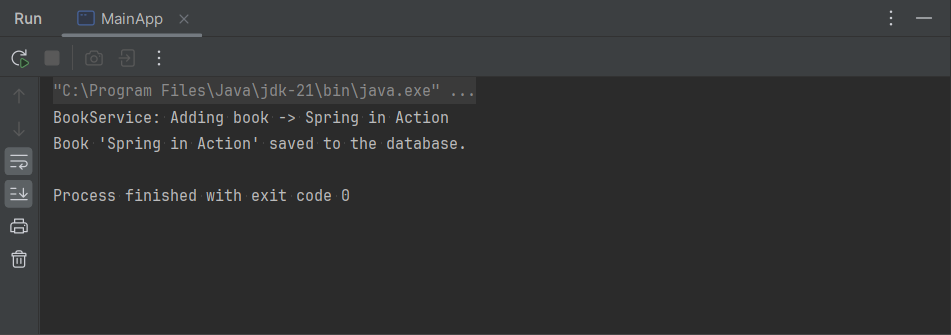
### ***BookRepository.java:***

| package com.library.repository;  public class BookRepository {  public void saveBook(String bookName) {  System.out.println("Book '" + bookName + "' is saved.");  } } |
| --- |

### ***MainApp.java:***

| package com.library;  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) {    ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");   BookService bookService = (BookService) context.getBean("bookService");   bookService.addBook("Spring in Action");  } } |
| --- |

***Output Screenshot:***

******

# **Exercise 2: Implementing Dependency Injection**

## **Scenario:** In the library management application, you need to manage the dependencies between the BookService and BookRepository classes using Spring's IoC and DI.

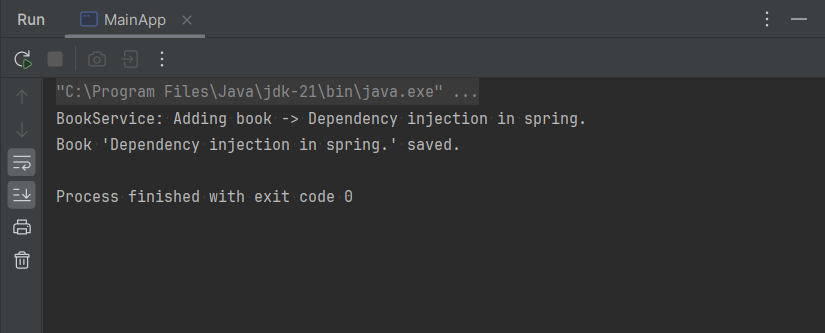
***pom.xml:***

| ***<?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">   <!-- Repository Bean -->  <bean id="bookRepository" class="com.library.repository.BookRepository" />   <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  public void setBookRepository(BookRepository bookRepository) {  this.bookRepository = bookRepository;  }   public void addBook(String bookName) {  System.out.println("BookService: Adding book -> " + bookName);  bookRepository.saveBook(bookName);  } }*** |
| --- |

***Output Screenshot:***

******

# 

# 

# 

# 

# 

# 

# 

# 

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

## **Scenario:** In the library management application, you need to manage the dependencies between the BookService and BookRepository classes using Spring's IoC and DI.

### ***pom.xml:*** *Included dependencies for Spring Context, Spring AOP, and Spring WebMVC and configured it for the maven compiler plugin for java version 1.8 .*

| <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>1.0-SNAPSHOT</version>   <properties>  <java.version>21</java.version>  </properties>   <dependencies>  *<!-- Spring Core-->*  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-context</artifactId>  <version>5.3.20</version>  </dependency>   *<!-- Spring AOP -->*  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-aop</artifactId>  <version>5.3.20</version>  </dependency>   *<!-- Spring Web MVC -->*  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-webmvc</artifactId>  <version>5.3.20</version>  </dependency>   <dependency>  <groupId>jakarta.servlet</groupId>  <artifactId>jakarta.servlet-api</artifactId>  <version>5.0.0</version>  <scope>provided</scope>  </dependency>  </dependencies>  <build>  <plugins>  <plugin>  <groupId>org.apache.maven.plugins</groupId>  <artifactId>maven-compiler-plugin</artifactId>  <version>3.11.0</version>  <configuration>  <source>1.8</source>  <target>1.8</target>  </configuration>  </plugin>  </plugins>  </build> </project> |
| --- |

### 