1. **Configuring a Basic Spring Application**

**Code:**

BookRepository.java

package com.library.repository;

public class BookRepository {

public void saveBook(String title) {

System.***out***.println("Saving book: " + title);

}

}

BookService.java

package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

private BookRepository bookRepository;

// Setter Injection

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

public void addBook(String title) {

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

bookRepository.saveBook(title);

}

}

Main.java

package com.library;

import com.library.service.BookService;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class Main {

*@SuppressWarnings*("resource")

public static void main(String[] args) {

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

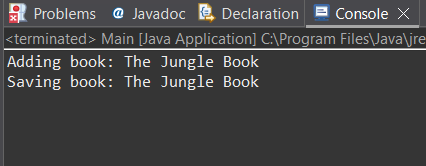
BookService bookService = (BookService) context.getBean("bookService");

bookService.addBook("The Jungle Book");

}

}

**Output:**



1. **Implementing Dependency Injection**

**Code:**

BookRepository.java

package com.library.repository;

import org.springframework.stereotype.Repository;

*@Repository*

public class BookRepository {

public void saveBook(String title) {

System.***out***.println("Saving book: " + title);

}

}

BookService.java

package com.library.service;

import com.library.repository.BookRepository;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

*@Service*

public class BookService {

*@Autowired*

private BookRepository bookRepository;

public void addBook(String title) {

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

bookRepository.saveBook(title);

}

}

Main.java

package com.library;

import com.library.service.BookService;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class Main {

*@SuppressWarnings*("resource")

public static void main(String[] args) {

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

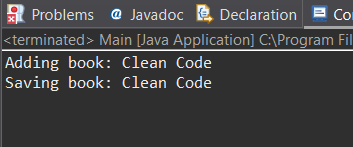
BookService bookService = (BookService) context.getBean("bookService");

bookService.addBook("Life of Pie");

}

}

**Output:**



**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>LibraryManagement</groupId>

<artifactId>LibraryManagement</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>LibraryManagement</name>

<dependencies>

<!-- Spring Context -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.2.22.RELEASE</version>

</dependency>

<!-- Spring AOP -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aop</artifactId>

<version>5.2.22.RELEASE</version>

</dependency>

<!-- Spring Web MVC -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>5.2.22.RELEASE</version>

</dependency>

<!-- Optional: Logging bridge (recommended) -->

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-api</artifactId>

<version>1.7.36</version>

</dependency>

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-simple</artifactId>

<version>1.7.36</version>

</dependency>

</dependencies>

<build>

<plugins>

<!-- Maven Compiler Plugin -->

<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>

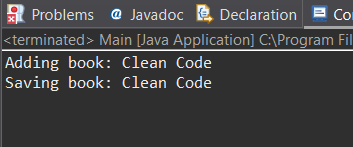
</plugin>

</plugins>

</build>

</project>

**Output:**



**5) Configuring the Spring IoC Container**

**Code:**

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"

xmlns:context="http://www.springframework.org/schema/context"

xsi:schemaLocation="

http://www.springframework.org/schema/beans

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

http://www.springframework.org/schema/context

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

<!-- Enable @Component and @Autowired scanning -->

<context:component-scan base-package="com.library" />

<!-- BookRepository Bean -->

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

<!-- BookService 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;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

*@Service*

public class BookService {

*@Autowired*

private BookRepository bookRepository;

// Setter for dependency injection

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

public void addBook(String title) {

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

bookRepository.saveBook(title);

}

}

BookRepository.java

package com.library.repository;

import org.springframework.stereotype.Repository;

*@Repository*

public class BookRepository {

public void saveBook(String title) {

System.***out***.println("Saving book: " + title);

}

}

Main.java

package com.library;

import com.library.service.BookService;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class Main {

*@SuppressWarnings*("resource")

public static void main(String[] args) {

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

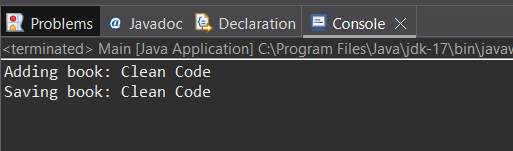
BookService bookService = context.getBean(BookService.class);

bookService.addBook("Clean Code");

}

}

**Output:**



**7) Implementing Constructor and Setter Injection**

**Code:**

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"

xmlns:context="http://www.springframework.org/schema/context"

xsi:schemaLocation="

http://www.springframework.org/schema/beans

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

http://www.springframework.org/schema/context

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

<!-- Enable @Component and @Autowired scanning -->

<context:component-scan base-package="com.library" />

<!-- BookRepository Bean -->

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

<!-- BookService Bean with Dependency Injection -->

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

<constructor-arg ref="bookRepository"/>

</bean>

</beans>

BookService.java

package com.library.service;

import com.library.repository.BookRepository;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

*@Service*

public class BookService {

*@Autowired*

private BookRepository bookRepository;

// Constructor injection

public BookService(BookRepository bookRepository) {

this.setBookRepository(bookRepository);

}

// Setter for dependency injection

public void setBookRepository(BookRepository bookRepository) {

this.bookRepository = bookRepository;

}

public void addBook(String title) {

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

bookRepository.saveBook(title);

}

}

BookRepository.java

package com.library.repository;

import org.springframework.stereotype.Repository;

*@Repository*

public class BookRepository {

public void saveBook(String title) {

System.***out***.println("Saving book: " + title);

}

}

Main.java

package com.library;

import com.library.service.BookService;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class Main {

*@SuppressWarnings*("resource")

public static void main(String[] args) {

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

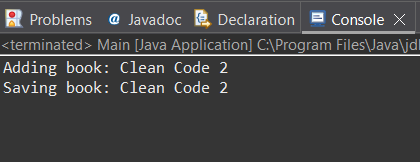
BookService bookService = context.getBean(BookService.class);

bookService.addBook("Clean Code 2");

}

}

**Output:**



**9) Creating a Spring Boot Application**

**Code:**

application.properties

spring.application.name=LibraryManagementWeb

# H2 Database Configuration

spring.datasource.url=jdbc:h2:mem:librarydb

spring.datasource.driverClassName=org.h2.Driver

spring.datasource.username=sa

spring.datasource.password=

spring.jpa.database-platform=org.hibernate.dialect.H2Dialect

spring.h2.console.enabled=true

spring.jpa.hibernate.ddl-auto=update

Book.java

package com.library.model;

import jakarta.persistence.\*;

*@Entity*

public class Book {

*@Id*

*@GeneratedValue*(strategy = *GenerationType*.***IDENTITY***)

private Long id;

private String title;

private String author;

private int publicationYear;

// Getters and setters

public Long getId() { return id; }

public void setId(Long id) { this.id = id; }

public String getTitle() { return title; }

public void setTitle(String title) { this.title = title; }

public String getAuthor() { return author; }

public void setAuthor(String author) { this.author = author; }

public int getPublicationYear() { return publicationYear; }

public void setPublicationYear(int publicationYear) { this.publicationYear = publicationYear; }

}

BookRepository.java

package com.library.repository;

import com.library.model.Book;

import org.springframework.data.jpa.repository.JpaRepository;

public interface BookRepository extends JpaRepository<Book, Long> {

}

BookController.java

package com.library.controller;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.\*;

import com.library.model.Book;

import com.library.repository.BookRepository;

import java.util.List;

*@RestController*

*@RequestMapping*("/books")

public class BookController {

*@Autowired*

private BookRepository bookRepository;

*@GetMapping*

public List<Book> getAllBooks() {

return bookRepository.findAll();

}

*@PostMapping*

public Book createBook(*@RequestBody* Book book) {

return bookRepository.save(book);

}

*@GetMapping*("/{id}")

public Book getBook(*@PathVariable* Long id) {

return bookRepository.findById(id).orElse(null);

}

*@PutMapping*("/{id}")

public Book updateBook(*@PathVariable* Long id, *@RequestBody* Book updatedBook) {

return bookRepository.findById(id).map(book -> {

book.setTitle(updatedBook.getTitle());

book.setAuthor(updatedBook.getAuthor());

book.setPublicationYear(updatedBook.getPublicationYear());

return bookRepository.save(book);

}).orElse(null);

}

*@DeleteMapping*("/{id}")

public void deleteBook(*@PathVariable* Long id) {

bookRepository.deleteById(id);

}

}

LibraryManagementWebApplication.java

package com.library;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

*@SpringBootApplication*

public class LibraryManagementWebApplication {

public static void main(String[] args) {

SpringApplication.*run*(LibraryManagementWebApplication.class, args);

}

}

**Output:**

