**Exercise 1: Configuring a Basic Spring Application (Mandatory)**

**POM.xml**

<dependency>

            <groupId>org.springframework</groupId>

            <artifactId>spring-context</artifactId>

            <version>5.3.34</version>

        </dependency>

**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.service*;

import *com.library.repository.BookRepository*;

*public* *class* BookService {

*private* BookRepository bookRepository;

*public* void setBookRepository(BookRepository bookRepository) {

        this.bookRepository = bookRepository;

    }

*public* void listBooks() {

        System.out.println("BookService is calling BookRepository!");

        bookRepository.displayBooks();

    }

}

**BookRepository.java**

package com.library.repository;

public class BookRepository {

      public void displayBooks() {

        System.out.println("Displaying list of books from the repository!");

    }

}

**LibraryManagementApplication.java**

package com.library;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.library.service.BookService;

public class LibraryManagementApplication {

    public static void main(String[] args) {

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

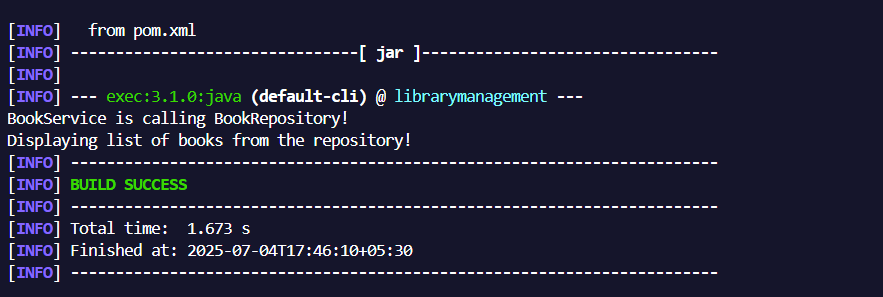
        BookService bookService = context.getBean("bookService", BookService.class);

        bookService.listBooks();

    }

}

**Output:**

****

**Exercise 2: Implementing Dependency Injection (Mandatory)**

**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.service*;

import *com.library.repository.BookRepository*;

*public* *class* BookService {

*private* BookRepository bookRepository;

*public* void setBookRepository(BookRepository bookRepository) {

        this.bookRepository = bookRepository;

    }

*public* void listBooks() {

        System.out.println("BookService is calling BookRepository!");

        bookRepository.displayBooks();

    }

}

**BookRepository.java**

package com.library.repository;

public class BookRepository {

      public void displayBooks() {

        System.out.println("Displaying list of books from the repository!");

    }

}

**LibraryManagementApplication.java**

package com.library;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.library.service.BookService;

public class LibraryManagementApplication {

    public static void main(String[] args) {

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

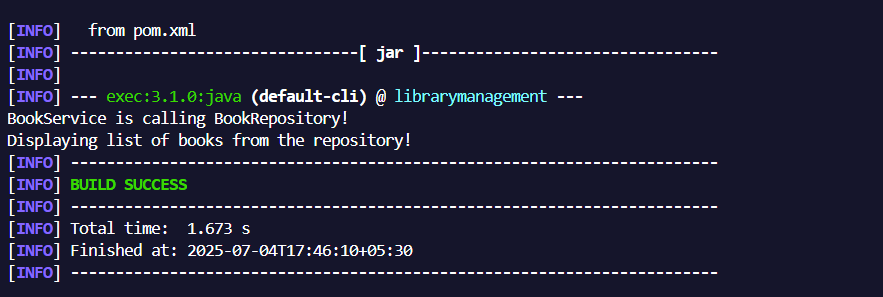
        BookService bookService = context.getBean("bookService", BookService.class);

        bookService.listBooks();

    }

}

**Output:**

****

**Exercise 3: Implementing Logging with Spring AOP**

**POM.xml**

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aop</artifactId>

<version>5.3.34</version>

</dependency>

<dependency>

<groupId>org.aspectj</groupId>

<artifactId>aspectjweaver</artifactId>

<version>1.9.21</version>

</dependency>

**LoggingAspect.java**

package com.library.aspect;

import org.aspectj.lang.ProceedingJoinPoint;

import org.aspectj.lang.annotation.Around;

import org.aspectj.lang.annotation.Aspect;

@Aspect

public class LoggingAspect {

@Around("execution(\* com.library.service.\*.\*(..))")

public Object logExecutionTime(ProceedingJoinPoint joinPoint) throws Throwable {

long start = System.currentTimeMillis();

Object result = joinPoint.proceed();

long end = System.currentTimeMillis();

System.out.println("Execution Time of " + joinPoint.getSignature() + " : " + (end - start) + " ms");

return result;

}

}

**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:aop="http://www.springframework.org/schema/aop"

xsi:schemaLocation="

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

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

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

http://www.springframework.org/schema/aop/spring-aop.xsd">

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

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

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

</bean>

<bean id="loggingAspect" class="com.library.aspect.LoggingAspect" />

<aop:aspectj-autoproxy />

</beans>

**LibraryManagementApplication.java**

package com.library;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.library.service.BookService;

public class LibraryManagementApplication {

    public static void main(String[] args) {

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

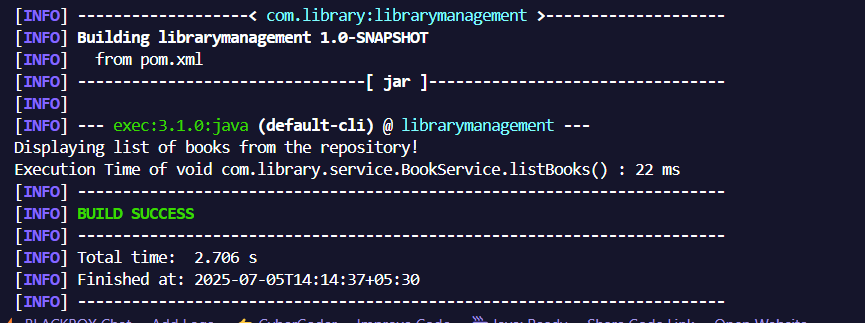
        BookService bookService = context.getBean("bookService", BookService.class);

        bookService.listBooks();

    }

}

**Output:**



**Exercise 4: Creating and Configuring a Maven Project (Mandatory)**

**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

                             http://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>

    <packaging>jar</packaging>

    <name>LibraryManagement</name>

    <properties>

        <maven.compiler.source>1.8</maven.compiler.source>

        <maven.compiler.target>1.8</maven.compiler.target>

    </properties>

    <dependencies>

        <dependency>

            <groupId>org.springframework</groupId>

            <artifactId>spring-context</artifactId>

            <version>5.3.34</version>

        </dependency>

        <dependency>

            <groupId>org.springframework</groupId>

            <artifactId>spring-aop</artifactId>

            <version>5.3.34</version>

        </dependency>

        <dependency>

            <groupId>org.aspectj</groupId>

            <artifactId>aspectjweaver</artifactId>

            <version>1.9.21</version>

        </dependency>

        <dependency>

            <groupId>org.springframework</groupId>

            <artifactId>spring-webmvc</artifactId>

            <version>5.3.34</version>

        </dependency>

        <dependency>

            <groupId>junit</groupId>

            <artifactId>junit</artifactId>

            <version>4.13.2</version>

            <scope>test</scope>

        </dependency>

    </dependencies>

**Maven Plugins**

    <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>

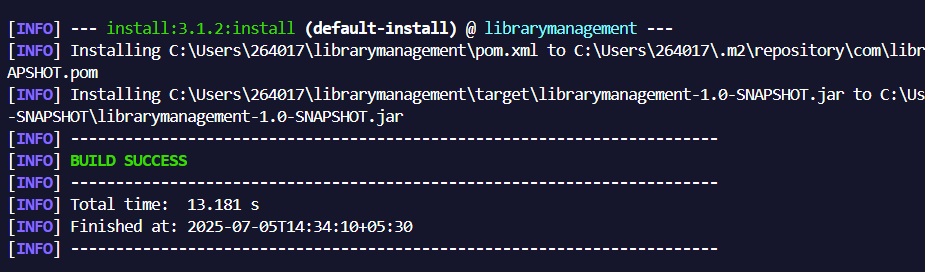
                </configuration>

            </plugin>

        </plugins>

    </build>

</project>

**Output**

**Exercise 5: Configuring the Spring IoC Container (Additional Importanat)**

**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:aop="http://www.springframework.org/schema/aop"

xsi:schemaLocation="

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

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

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

http://www.springframework.org/schema/aop/spring-aop.xsd">

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

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

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

</bean>

<bean id="loggingAspect" class="com.library.aspect.LoggingAspect" />

<aop:aspectj-autoproxy />

</beans>

**BookService.java**

package *com.library.service*;

import *com.library.repository.BookRepository*;

*public* *class* BookService {

*private* BookRepository bookRepository;

*public* void setBookRepository(BookRepository bookRepository) {

        this.bookRepository = bookRepository;

    }

*public* void listBooks() {

        System.out.println("BookService is calling BookRepository!");

        bookRepository.displayBooks();

    }

}

**BookRepository.java**

package com.library.repository;

public class BookRepository {

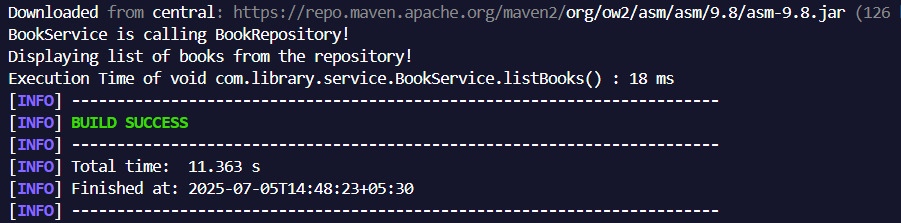
      public void displayBooks() {

        System.out.println("Displaying list of books from the repository!");

    }

}

**Output**

****

**Exercise 6: Configuring Beans with Annotations**

**applicationContext.xml**

<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"

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

       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

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

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

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

</beans>

**BookService.java**

package com.library.service;

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

import org.springframework.stereotype.Service;

import com.library.repository.BookRepository;

@Service

public class BookService {

    @Autowired

    private BookRepository bookRepository;

    public void listBooks() {

        System.out.println("BookService is calling BookRepository!");

        bookRepository.displayBooks();

    }

}

**BookRepository.java**

package com.library.repository;

import org.springframework.stereotype.Repository;

@Repository

public class BookRepository {

    public void displayBooks() {

        System.out.println("Displaying list of books from the repository!");

    }

}

**LibraryManagement.java**

package com.library;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.library.service.BookService;

public class LibraryManagementApplication {

    public static void main(String[] args) {

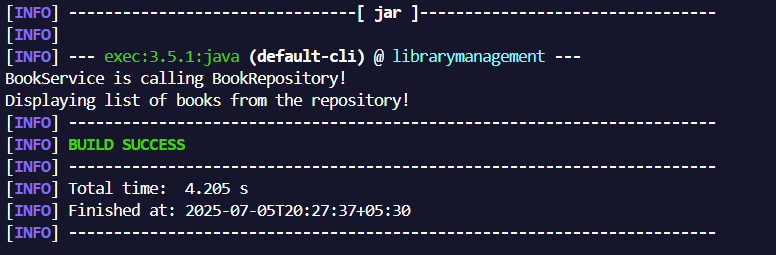
        ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");

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

        bookService.listBooks();

    }

}

**Output**

**Exercise 7: Implementing Constructor and Setter Injection (Additional Important)**

**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">

        <constructor-arg index="0" value="Elsa" />

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

    </bean>

</beans>

**BookService.java**

package com.library.service;

import com.library.repository.BookRepository;

public class BookService {

    private String managerName;

    private BookRepository bookRepository;

    public BookService(String managerName) {

        this.managerName = managerName;

    }

    public void setBookRepository(BookRepository bookRepository) {

        this.bookRepository = bookRepository;

    }

    public void listBooks() {

        System.out.println("The Manager: " + managerName);

        System.out.println("BookService is calling BookRepository...");

        bookRepository.displayBooks();

    }

}

**LibraryManagement.java**

package com.library;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.library.service.BookService;

public class LibraryManagementApplication {

    public static void main(String[] args) {

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

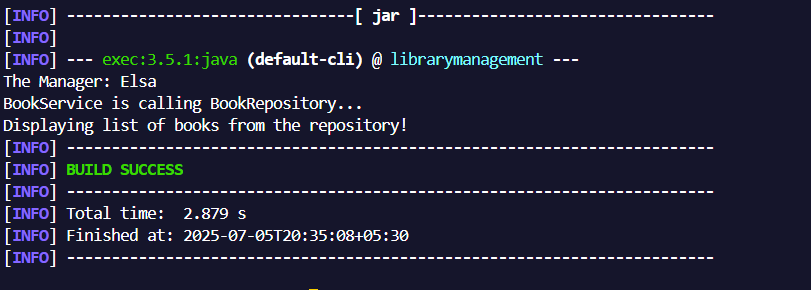
        BookService service = context.getBean("bookService", BookService.class);

        service.listBooks();

    }

}

**Output**

****

**Exercise 8: Implementing Basic AOP with Spring**

**LoggingAspect.java**

package com.library.aspect;

import org.aspectj.lang.JoinPoint;

import org.aspectj.lang.annotation.After;

import org.aspectj.lang.annotation.Aspect;

import org.aspectj.lang.annotation.Before;

@Aspect

public class LoggingAspect {

    @Before("execution(\* com.library.service.\*.\*(..))")

    public void logBefore(JoinPoint joinPoint) {

        System.out.println("Before Method Execution: " + joinPoint.getSignature().getName());

    }

    @After("execution(\* com.library.service.\*.\*(..))")

    public void logAfter(JoinPoint joinPoint) {

        System.out.println("After Method Execution: " + joinPoint.getSignature().getName());

    }

}

**applicationContext.xml**

<beans xmlns="http://www.springframework.org/schema/beans"

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

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

       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

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

           http://www.springframework.org/schema/aop/spring-aop.xsd

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

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

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

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

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

        <constructor-arg index="0" value="Elsa" />

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

    </bean>

    <bean id="loggingAspect" class="com.library.aspect.LoggingAspect" />

    <aop:aspectj-autoproxy />

</beans>

**LibraryManagement.java**

package com.library;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.library.service.BookService;

public class LibraryManagementApplication {

    public static void main(String[] args) {

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

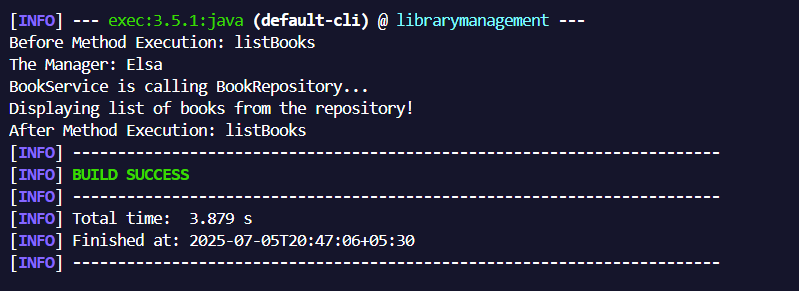
        BookService service = context.getBean("bookService", BookService.class);

        service.listBooks();

    }

}

**Output**

****

**Exercise 9: Creating a Spring Boot Application (Additional Important)**

**application.properties**

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.jpa.hibernate.ddl-auto=update

spring.h2.console.enabled=true

server.port=9090

**BookRepository.java**

package *com.library.librarymanagement.repository*;

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

import *com.library.librarymanagement.entity.Book*;

*public* *interface* BookRepository *extends* JpaRepository<Book, Long> {

}

**BookController.java**

package *com.library.librarymanagement.controller*;

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

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

import *java.util.List*;

import *com.library.librarymanagement.entity.Book*;

import *com.library.librarymanagement.repository.BookRepository*;

@RestController

@RequestMapping("/books")

*public* *class* BookController {

    @Autowired

*private* BookRepository bookRepository;

    @GetMapping

*public* List<Book> getAllBooks() {

        return bookRepository.findAll();

    }

    @GetMapping("/{id}")

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

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

    }

    @PostMapping

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

        return bookRepository.save(book);

    }

    @PutMapping("/{id}")

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

        book.setId(id);

        return bookRepository.save(book);

    }

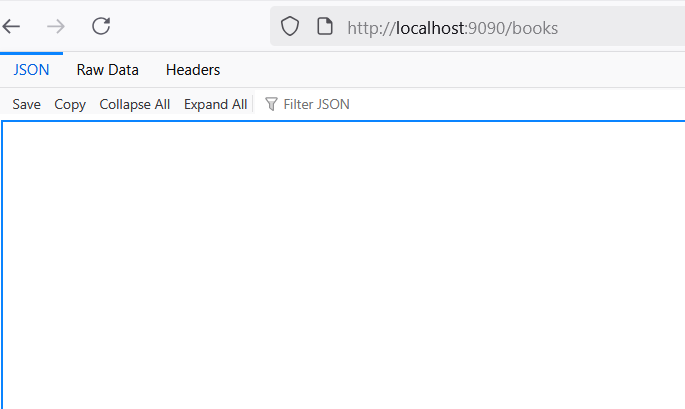
    @DeleteMapping("/{id}")

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

        bookRepository.deleteById(id);

    }

}

**Output**