

## 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.

#### Steps:

##### 1. Set Up a Spring Project:

- Create a Maven project named **LibraryManagement**.
- Add Spring Core dependencies in the **pom.xml** file.

##### 2. Configure the Application Context:

- Create an XML configuration file named **applicationContext.xml** in the **src/main/resources** directory.
- Define beans for **BookService** and **BookRepository** in the XML file.

##### 3. Define Service and Repository Classes:

- Create a package **com.library.service** and add a class **BookService**.
- Create a package **com.library.repository** and add a class **BookRepository**.

##### 4. Run the Application:

- Create a main class to load the Spring context and test the configuration.

#### Code :

##### **pom.xml :**

```
<?xml version="1.0" encoding="UTF-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
    http://maven.apache.org/xsd/maven-4.0.0.xsd">

  <modelVersion>4.0.0</modelVersion>

  <groupId>com.springcoreandmaven</groupId>
```

<artifactId>LibraryManagement</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>LibraryManagement</name>

<description>A simple LibraryManagement.</description>

<!-- FIXME change it to the project's website -->

<url>http://www.example.com</url>

<properties>

    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

    <maven.compiler.source>8</maven.compiler.source>

    <maven.compiler.target>8</maven.compiler.target>

</properties>

<dependencies>

    <dependency>

        <groupId>junit</groupId>

        <artifactId>junit</artifactId>

        <version>3.8.1</version>

    </dependency>

    <dependency>

        <groupId>org.springframework</groupId>

        <artifactId>spring-context</artifactId>

        <version>5.3.32</version>

    </dependency>

    <dependency>

        <groupId>org.springframework</groupId>

        <artifactId>spring-aop</artifactId>

        <version>5.3.30</version>

    </dependency>

```
<!-- Spring Web MVC -->
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-webmvc</artifactId>
  <version>5.3.30</version>
</dependency>
</dependencies>

<build>
  <pluginManagement><!-- lock down plugins versions to avoid using Maven defaults (may
be moved to parent pom) -->
    <plugins>
      <plugin>
        <artifactId>maven-clean-plugin</artifactId>
        <version>3.4.0</version>
      </plugin>
      <plugin>
        <artifactId>maven-site-plugin</artifactId>
        <version>3.12.1</version>
      </plugin>
      <plugin>
        <artifactId>maven-project-info-reports-plugin</artifactId>
        <version>3.6.1</version>
      </plugin>
      <!-- see http://maven.apache.org/ref/current/maven-core/default-
bindings.html#Plugin\_bindings\_for\_jar\_packaging -->
      <plugin>
        <artifactId>maven-resources-plugin</artifactId>
        <version>3.3.1</version>
      </plugin>
    </plugins>
  </pluginManagement>
</build>
```

```

<plugin>
  <artifactId>maven-compiler-plugin</artifactId>
  <version>3.13.0</version>
  <configuration>
    <source>1.8</source>
    <target>1.8</target>
  </configuration>
</plugin>
<plugin>
  <artifactId>maven-surefire-plugin</artifactId>
  <version>3.3.0</version>
</plugin>
<plugin>
  <artifactId>maven-jar-plugin</artifactId>
  <version>3.4.2</version>
</plugin>
<plugin>
  <artifactId>maven-install-plugin</artifactId>
  <version>3.1.2</version>
</plugin>
<plugin>
  <artifactId>maven-deploy-plugin</artifactId>
  <version>3.1.2</version>
</plugin>
</plugins>
</pluginManagement>
</build>

<reporting>
  <plugins>

```

```

    <plugin>
      <artifactId>maven-project-info-reports-plugin</artifactId>
    </plugin>
  </plugins>
</reporting>
</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">

  <!-- BookRepository Bean -->

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

  <!-- BookService Bean and Inject BookRepository -->

  <bean id="bookService"
        class="com.springcoreandmaven.LibraryManagement.service.BookService">
    <property name="bookRepository" ref="bookRepository" />
  </bean>

</beans>

```

### **BookService.java :**

```

package com.springcoreandmaven.LibraryManagement.service;

import com.springcoreandmaven.LibraryManagement.repository.BookRepository;

```

```

public class BookService {
    private BookRepository bookRepository;

    // Setter for DI
    public void setBookRepository(BookRepository bookRepository) {
        this.bookRepository = bookRepository;
    }

    public void addBook(String name) {
        System.out.println("BookService: Adding book...");
        bookRepository.save(name);
    }
}

```

### **BookRepository.java :**

```

package com.springcoreandmaven.LibraryManagement.repository;

```

```

public class BookRepository {
    public void save(String bookName) {
        System.out.println("Saving book: " + bookName);
    }
}

```

### **App.java :**

```

package com.springcoreandmaven.LibraryManagement;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

```

```

import com.springcoreandmaven.LibraryManagement.service.BookService;

public class App {

    public static void main(String[] args) {

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

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

        service.addBook("Clean Code");

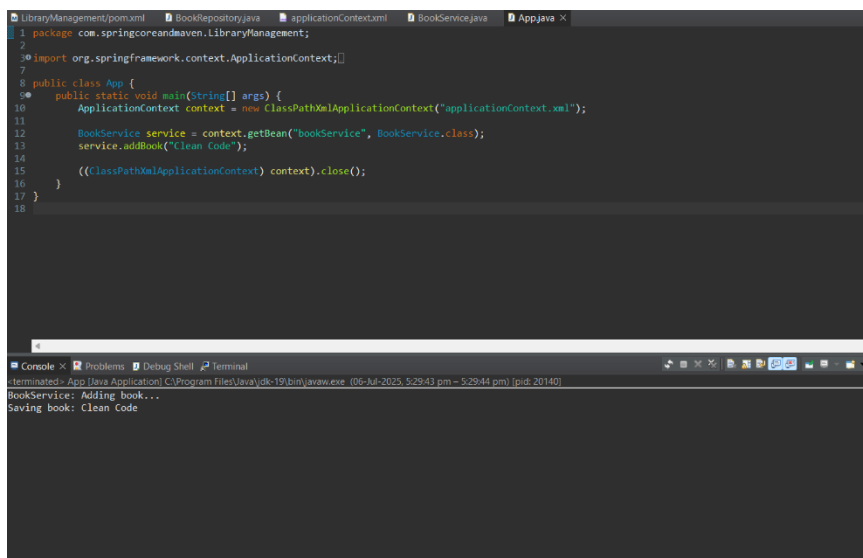
        ((ClassPathXmlApplicationContext) context).close();

    }

}

```

## Output :



The screenshot shows an IDE with several tabs open: LibraryManagement/pom.xml, BookRepository.java, applicationContext.xml, BookService.java, and App.java. The App.java tab is active, displaying the following code:

```

1 package com.springcoreandmaven.LibraryManagement;
2
3 import org.springframework.context.ApplicationContext;
4
5 public class App {
6     public static void main(String[] args) {
7         ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
8         BookService service = context.getBean("bookService", BookService.class);
9         service.addBook("Clean Code");
10        ((ClassPathXmlApplicationContext) context).close();
11    }
12 }

```

Below the code editor, the Console window shows the output of the application:

```

-terminated> App [Java Application] C:\Program Files\Java\jdk-19\bin\java.exe (06-Jul-2025, 5:29:43 pm - 5:29:44 pm) [pid: 20140]
BookService: Adding book...
Saving book: Clean Code

```

## 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.

### Steps:

#### 1. Modify the XML Configuration:

- Update **applicationContext.xml** to wire **BookRepository** into **BookService**.

#### 2. Update the **BookService** Class:

- Ensure that **BookService** class has a setter method for **BookRepository**.

#### 3. Test the Configuration:

- Run the **LibraryManagementApplication** main class to verify the dependency injection.

### Code :

```
<?xml version="1.0" encoding="UTF-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
http://maven.apache.org/xsd/maven-4.0.0.xsd">
```

```
  <modelVersion>4.0.0</modelVersion>
```

```
  <groupId>com.springcoreandmaven</groupId>
```

```
  <artifactId>ImplementingDependencyInjection</artifactId>
```

```
  <version>0.0.1-SNAPSHOT</version>
```

```
  <name>ImplementingDependencyInjection</name>
```

```
  <description>A simple ImplementingDependencyInjection.</description>
```

```
  <!-- FIXME change it to the project's website -->
```

```
  <url>http://www.example.com</url>
```



```
<properties>
  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  <maven.compiler.source>8</maven.compiler.source>
  <maven.compiler.target>8</maven.compiler.target>
</properties>
```

```
<dependencies>
  <dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>3.8.1</version>
  </dependency>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>5.3.30</version>
  </dependency>
</dependencies>
```

```
<build>
  <pluginManagement><!-- lock down plugins versions to avoid using Maven defaults (may
be moved to parent pom) -->
    <plugins>
      <plugin>
        <artifactId>maven-clean-plugin</artifactId>
        <version>3.4.0</version>
      </plugin>
      <plugin>
        <artifactId>maven-site-plugin</artifactId>
        <version>3.12.1</version>
      </plugin>
    </plugins>
  </pluginManagement>
</build>
```

```
<plugin>
  <artifactId>maven-project-info-reports-plugin</artifactId>
  <version>3.6.1</version>
</plugin>

<!-- see http://maven.apache.org/ref/current/maven-core/default-
bindings.html#Plugin_bindings_for_jar_packaging -->
<plugin>
  <artifactId>maven-resources-plugin</artifactId>
  <version>3.3.1</version>
</plugin>
<plugin>
  <artifactId>maven-compiler-plugin</artifactId>
  <version>3.13.0</version>
</plugin>
<plugin>
  <artifactId>maven-surefire-plugin</artifactId>
  <version>3.3.0</version>
</plugin>
<plugin>
  <artifactId>maven-jar-plugin</artifactId>
  <version>3.4.2</version>
</plugin>
<plugin>
  <artifactId>maven-install-plugin</artifactId>
  <version>3.1.2</version>
</plugin>
<plugin>
  <artifactId>maven-deploy-plugin</artifactId>
  <version>3.1.2</version>
</plugin>
</plugins>
```

```

    </pluginManagement>
</build>

<reporting>
  <plugins>
    <plugin>
      <artifactId>maven-project-info-reports-plugin</artifactId>
    </plugin>
  </plugins>
</reporting>
</project>

```

### **applicationContext.java :**

```

<?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">

  <!-- Define BookRepository Bean -->

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

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

  <bean id="bookService"
class="com.springcoreandmaven.ImplementingDependencyInjection.service.BookService">
    <property name="bookRepository" ref="bookRepository" />
  </bean>

```

</beans>

### **BookService.java :**

```
package com.springcoreandmaven.ImplementingDependencyInjection.service;

import
com.springcoreandmaven.ImplementingDependencyInjection.repository.BookRepository;

public class BookService {
    private BookRepository bookRepository;

    // Setter method for Spring to inject
    public void setBookRepository(BookRepository bookRepository) {
        this.bookRepository = bookRepository;
    }

    public void addBook(String name) {
        System.out.println("BookService: Adding book...");
        bookRepository.save(name);
    }
}
```

### **BookRepository.java :**

```
package com.springcoreandmaven.ImplementingDependencyInjection.repository;

public class BookRepository {
    public void save(String bookName) {
        System.out.println("Saving book: " + bookName);
    }
}
```

## App.java :

```
package com.springcoreandmaven.ImplementingDependencyInjection;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.springcoreandmaven.ImplementingDependencyInjection.service.BookService;

public class App {

    public static void main(String[] args) {

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

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

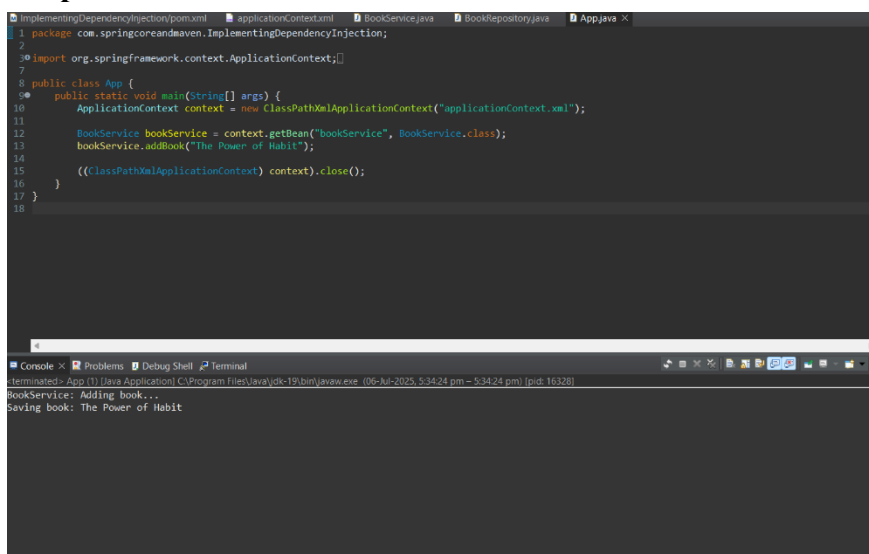
        bookService.addBook("The Power of Habit");

        ((ClassPathXmlApplicationContext) context).close();

    }

}
```

## Output :



The screenshot shows an IDE with several tabs open: 'ImplementingDependencyInjection/pom.xml', 'applicationContext.xml', 'BookService.java', 'BookRepository.java', and 'App.java'. The 'App.java' tab is active, displaying the code from the previous block. Below the code editor, the 'Console' tab is open, showing the output of the application. The output consists of two lines: 'BookService: Adding book...' and 'Saving book: The Power of Habit'. The console also shows a system message at the top: 'terminated> App (1) [Java Application] C:\Program Files\Java\jdk-19\bin\java.exe. (05-Jul-2025, 5:34:24 pm - 5:34:24 pm) (pid: 16328)'.

```
1 package com.springcoreandmaven.ImplementingDependencyInjection;
2
3 import org.springframework.context.ApplicationContext;
4
5 public class App {
6     public static void main(String[] args) {
7         ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
8         BookService bookService = context.getBean("bookService", BookService.class);
9         bookService.addBook("The Power of Habit");
10        ((ClassPathXmlApplicationContext) context).close();
11    }
12}

BookService: Adding book...
Saving book: The Power of Habit
```

## Exercise 4: Creating and Configuring a Maven Project

### Scenario:

You need to set up a new Maven project for the library management application and add Spring dependencies.

### Steps:

#### 1. Create a New Maven Project:

- Create a new Maven project named **LibraryManagement**.

#### 2. Add Spring Dependencies in pom.xml:

- Include dependencies for Spring Context, Spring AOP, and Spring WebMVC.

#### 3. Configure Maven Plugins:

- Configure the Maven Compiler Plugin for Java version 1.8 in the pom.xml file.

### Code :

#### pom.xml :

```
<?xml version="1.0" encoding="UTF-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
    http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>com.springcoreandmaven</groupId>
  <artifactId>LibraryManagement</artifactId>
  <version>0.0.1-SNAPSHOT</version>

  <name>LibraryManagement</name>
  <description>A simple LibraryManagement.</description>
  <!-- FIXME change it to the project's website -->
  <url>http://www.example.com</url>
```

```
<properties>
  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  <maven.compiler.source>8</maven.compiler.source>
  <maven.compiler.target>8</maven.compiler.target>
</properties>
```

```
<dependencies>
  <dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>3.8.1</version>
  </dependency>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>5.3.32</version>
  </dependency>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-aop</artifactId>
    <version>5.3.30</version>
  </dependency>

  <!-- Spring Web MVC -->
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-webmvc</artifactId>
    <version>5.3.30</version>
  </dependency>
```

</dependencies>

<build>

<pluginManagement><!-- lock down plugins versions to avoid using Maven defaults (may be moved to parent pom) -->

<plugins>

<plugin>

<artifactId>maven-clean-plugin</artifactId>

<version>3.4.0</version>

</plugin>

<plugin>

<artifactId>maven-site-plugin</artifactId>

<version>3.12.1</version>

</plugin>

<plugin>

<artifactId>maven-project-info-reports-plugin</artifactId>

<version>3.6.1</version>

</plugin>

<!-- see [http://maven.apache.org/ref/current/maven-core/default-bindings.html#Plugin\\_bindings\\_for\\_jar\\_packaging](http://maven.apache.org/ref/current/maven-core/default-bindings.html#Plugin_bindings_for_jar_packaging) -->

<plugin>

<artifactId>maven-resources-plugin</artifactId>

<version>3.3.1</version>

</plugin>

<plugin>

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

<version>3.13.0</version>

<configuration>

<source>1.8</source>

<target>1.8</target>

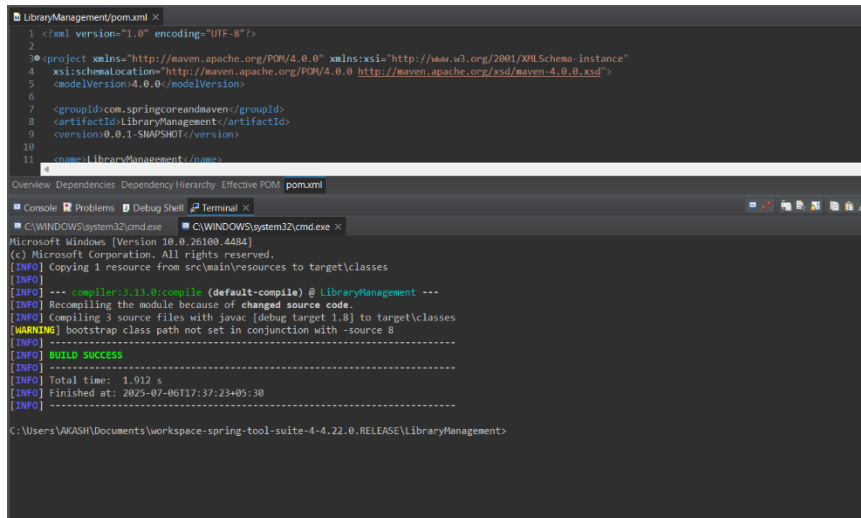
</configuration>



```
</plugin>
<plugin>
  <artifactId>maven-surefire-plugin</artifactId>
  <version>3.3.0</version>
</plugin>
<plugin>
  <artifactId>maven-jar-plugin</artifactId>
  <version>3.4.2</version>
</plugin>
<plugin>
  <artifactId>maven-install-plugin</artifactId>
  <version>3.1.2</version>
</plugin>
<plugin>
  <artifactId>maven-deploy-plugin</artifactId>
  <version>3.1.2</version>
</plugin>
</plugins>
</pluginManagement>
</build>

<reporting>
  <plugins>
    <plugin>
      <artifactId>maven-project-info-reports-plugin</artifactId>
    </plugin>
  </plugins>
</reporting>
</project>
```

## Output :



```
LibraryManagement/pom.xml x
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
5   <modelVersion>4.0.0</modelVersion>
6
7   <groupId>com.springcoreandmaven</groupId>
8   <artifactId>LibraryManagement</artifactId>
9   <version>0.0.1-SNAPSHOT</version>
10
11   <name>LibraryManagement</name>
</project>

Overview Dependencies Dependency Hierarchy Effective POM pom.xml
Console Problems Debug Shell Terminal x
C:\WINDOWS\system32\cmd.exe C:\WINDOWS\system32\cmd.exe x
Microsoft Windows [Version 10.0.26100.4484]
(c) Microsoft Corporation. All rights reserved.
[INFO] Copying 1 resource from src/main/resources to target/classes
[INFO]
[INFO] --- compiler:3.11.0:compile (default-compile) @ LibraryManagement ---
[INFO] Recompiling the module because of changed source code.
[INFO] Compiling 3 source files with javac [debug target 1.8] to target/classes
[WARNING] bootstrap class path not set in conjunction with -source 8
[INFO]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO]
[INFO] Total time: 1.912 s
[INFO] Finished at: 2025-07-06T17:37:23+05:30
[INFO]
C:\Users\AKASH\Documents\workspace-spring-tool-suite-4-4.22.0.RELEASE\LibraryManagement>
```