**Configuring the Spring IoC Container**

**Creating Spring Configuration File (Step 1):**

* Create an XML file named applicationContext.xml within the src/main/resources directory.
* This file will hold the bean definitions for the application components.

**Defining Beans in applicationContext.xml:**

* The solution demonstrates how to define beans for BookService and BookRepository.
  + <bean id="..."> Defines a bean with a specific id.
  + class="..." Specifies the class that the bean represents.
  + <property name="..."> Configures a property setter on the bean.
    - ref="..." References another bean by its id for injection.

**Updating the BookService Class (Step 2):**

* Ensure the BookService class has a setter method named setBookRepository that takes a BookRepository argument.
* Spring will utilize this setter during object creation to inject the dependency.

**Running the Application (Step 3):**

* Create a MainApp class to interact with the Spring IoC container.
* In MainApp.main:
  + Load the Spring context using ClassPathXmlApplicationContext("applicationContext.xml").
  + Retrieve the BookService bean by its id.
  + Call bookService.manageBooks() to test the configuration and dependency injection.

**Summary:**

* A Spring configuration file to manage application beans has established.
* The BookService receives its BookRepository dependency through injection.
* Running MainApp demonstrates successful configuration and interaction with Spring.

**Extra Notes:**

* While XML configuration is traditional, considering using Java based configuration is good for larger projects.
* This exercise showcases the fundamental concepts of Dependency Injection (DI) using Spring.