**Display Book Details - Autowiring**

Description

**Objective:**

To work with a Spring Core application using annotations, injecting values into the field, and autowiring concepts.

**Concept Explanation:**

1. Spring allows **injecting values into fields** of Spring-managed beans using **@Value** annotation.
2. **Autowiring** in Spring automatically wires beans together based on dependency types.

**Concept Implementation:**

1. The **@Value** annotation is used to inject values into the fields of the **Book** and **Order** classes.
2. **Autowiring** is implemented by using the **@Autowired** annotation above the books property in the **Order** class.
3. The **ApplicationConfig** class is annotated with **@ComponentScan** to enable component scanning, allowing Spring to detect and register the beans in the specified package.

**Display Book Details - Autowiring**

**Book**class with the below **private attributes**is provided as a part of code skeleton

|  |  |
| --- | --- |
| bookId | int |
| bookAuthor | String |
| bookTitle | String |
| price | double |

**Getter and setter** methods for all the above attributes are provided as a part of code skeleton. Use appropriate spring annotation above the **Book**class to denote the class as component. Use appropriate spring annotation above the attributes to assign values.

**Order**class with the below **private attributes**is provided as a part of code skeleton

|  |  |
| --- | --- |
| books | Book |
| quantity | int |

**Getter and setter** methods for the above attributes are provided as a part of code skeleton. Use appropriate spring annotation above the **Order**class to denote the class as component. Use appropriate spring annotation above the attribute "**quantity**" to assign values.

The **Book**object should be autowired above the property via annotations. Create a class called **ApplicationConfig** that has the required annotations for **scanning** and **registering** the bean definitions.

A method **public double calculateTotalPrice()**will be provided in the **Order**class as a part of code skeleton. This method is used to calculate the **totalPrice**for the books ordered and return the same to the main method.

A method **public void displayOrderDetails()**will be provided in the **Order**class as a part of code skeleton. This method is used to display the Invoice details as shown in the sample output.

**Driver**class with the below methods are provided as a part of code skeleton

* **public static Order loadBookDetails()**--> This method should fetch the **Order**object by loading **ApplicationConfig** class and return the same
* **public static void main(String[] args)**-->  Inside the main method invoke the **loadBookDetails**method and obtain the **Order**object.

**Design Constraints:**

* **Book,** **Order**and **ApplicationConfig**class should be public and should be present in**com.spring.app** package
* The className/Attribute Name/PackageName should be same as specified in the problem statement. Do not create any new packages.
* Do not create any XML configuration file. You need to use only annotations for configuration.

**Sample Output:**

Invoice Details:

Book Id:123

Book Title:Programming Using C

Author:Dennis M.Ritchie

Quantity:2

Price of a book:280.0

Total Bill Amount:560.0