APPLICATION DEVELOPMENT PROJECT USING JAVA

(CSE4171)

Assignment - 4

Topic: SpringBoot Starter, SpringBoot Starter Dependencies and Auto Configuration, Annotation-Based Java Configuration, Xml-Based Configuration, Loosely Coupled Dependency Injection

---

1. Write a Java Program to show Loosely Coupling using Spring XML configuration and Constructor-based Dependency Injection using the following:
    a. PaymentMethod Interface
    b. CreditCard Class implements the PaymentMethod Interface
    c. PayPal Class implements the PaymentMethod Interface
    d. ShoppingCart Class uses the PaymentMethod instance to process payments.
    e. App Class to run the application
2. Write a Java Program showing setter-based dependency injection with reference in XML Configuration using the following.
    a. EmailService class with a method sendEmail(String msg)()
    b. SMSService class with a method sendSMS(String msg)()
    c. NotificationService depends on both the above classes with setter methods
3. Write a Java Program to demonstrate annotation-based Java configuration
    a. UserRepository class with a method saveUser()
    b. UserService class depending on the above class with a setter method DI.
4. Write a Java Program to demonstrate a simple example using <context:component-scan> with annotation-based configuration in Spring.
    a. GreetingService Interface
    b. GreetingServiceImplement class implementing the above interface

5. Write POJO Java program to convert tightly coupled code into loosely coupled code
    a. Create a parent class A with a method display(). Create another class B that inherits class A and contains a method display(). Create a main class to call the display method.
    b. Create a class LightBulb with a method SwitchOn(). Create another class Switch that has an object of the LightBulb class and another method Operate(). Inside the Operate() method call the SwitchOn() method
6. Write a simple SpringBoot Project and add the Spring Web and Spring Boot Dev Tools dependencies. Execute the application.
7. Write a SpringBoot Program with the following:
    a. Create an Employee class with two instance variables, name and age.
    b. Add a parameterized constructor to set the data.
    c. Add an overridden toString() method to print the details.
8. How does Spring Boot use `@ConditionalOnClass` to trigger auto-configuration?
9. Explain the role of `@ConditionalOnProperty` in controlling auto-configuration behavior.
10. What is the significance of `@ConditionalOnBean` in Spring Boot's auto-configuration process?
11. What is the purpose of the `spring-boot-starter-parent` in a Spring Boot application, and how do we dive deeper into the structure of any dependency used within the project?