WEEK – 1

Implementing the Decorator pattern

Nipuna Amanapu

[namanapu@gitam.in](mailto:namanapu@gitam.in)

superset id : 6432842

**Exercise 5: Implementing the Decorator Pattern**

**Scenario:**

You are developing a notification system where notifications can be sent via multiple channels (e.g., Email, SMS). Use the Decorator Pattern to add functionalities dynamically.

**Steps:**

1. **Create a New Java Project:**
   * Create a new Java project named **DecoratorPatternExample**.
2. **Define Component Interface:**
   * Create an interface **Notifier** with a method **send()**.
3. **Implement Concrete Component:**
   * Create a class **EmailNotifier** that implements Notifier.
4. **Implement Decorator Classes:**
   * Create abstract decorator class **NotifierDecorator** that implements **Notifier** and holds a reference to a **Notifier** object.
   * Create concrete decorator classes like **SMSNotifierDecorator**, **SlackNotifierDecorator** that extend **NotifierDecorator**.
5. **Test the Decorator Implementation:**
   * Create a test class to demonstrate sending notifications via multiple channels using decorators.

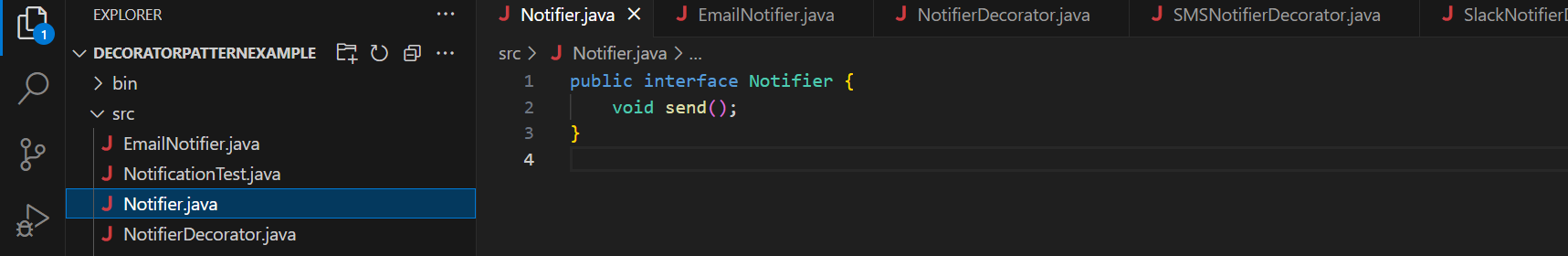
Solution:

1. Notifier.java

public interface Notifier {

void send();

}



2)EmailNotifier.java

public class EmailNotifier implements Notifier {

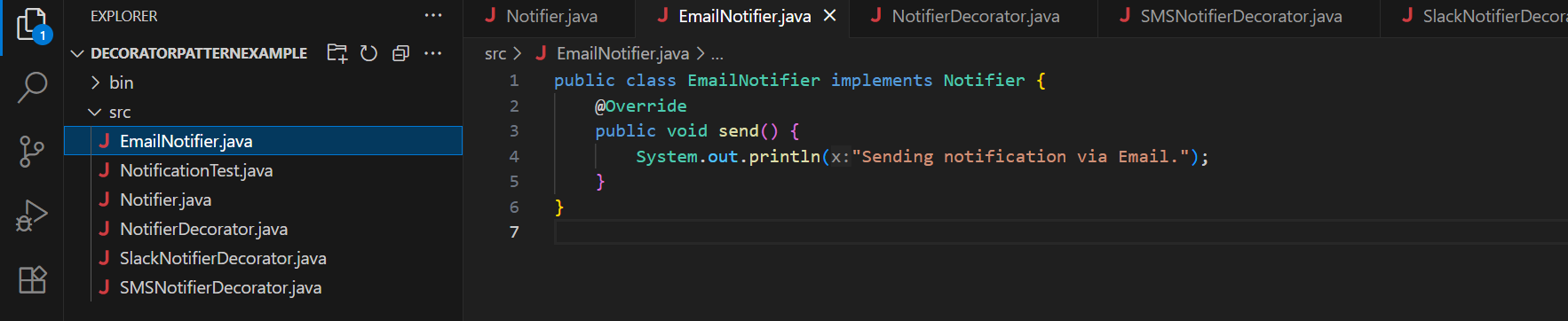
@Override

public void send() {

System.out.println("Sending notification via Email.");

}

}



3)NotifierDecorator.java

public abstract class NotifierDecorator implements Notifier {

protected Notifier wrappee;

public NotifierDecorator(Notifier notifier) {

this.wrappee = notifier;

}

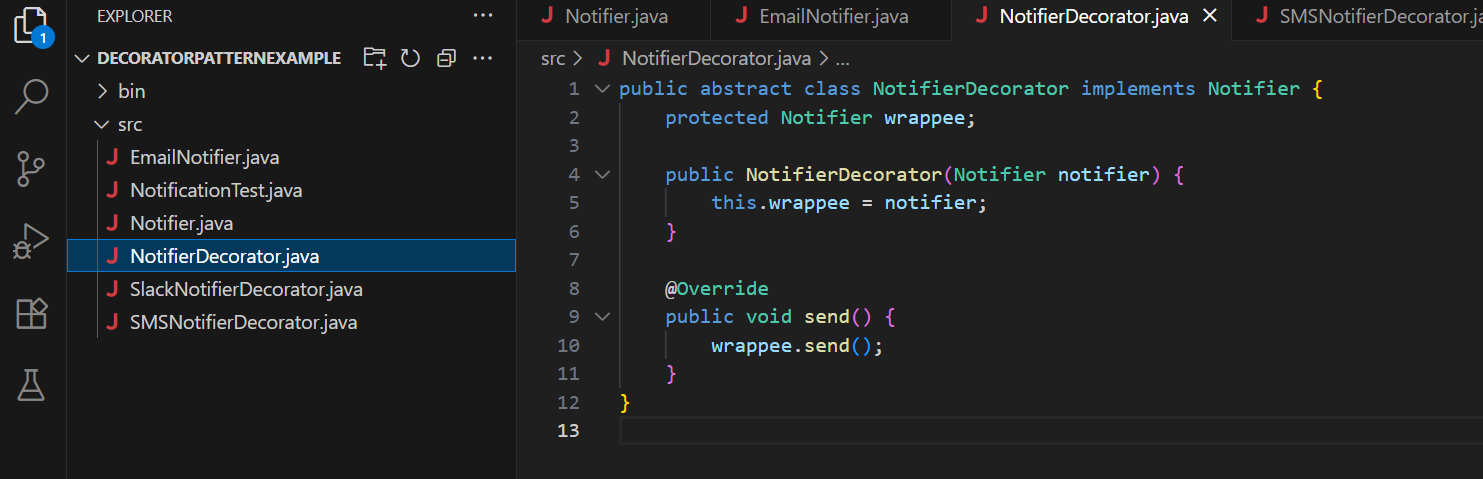
@Override

public void send() {

wrappee.send();

}

}



4) SMSNotifierDecorator.java

public class SMSNotifierDecorator extends NotifierDecorator {

public SMSNotifierDecorator(Notifier notifier) {

super(notifier);

}

@Override

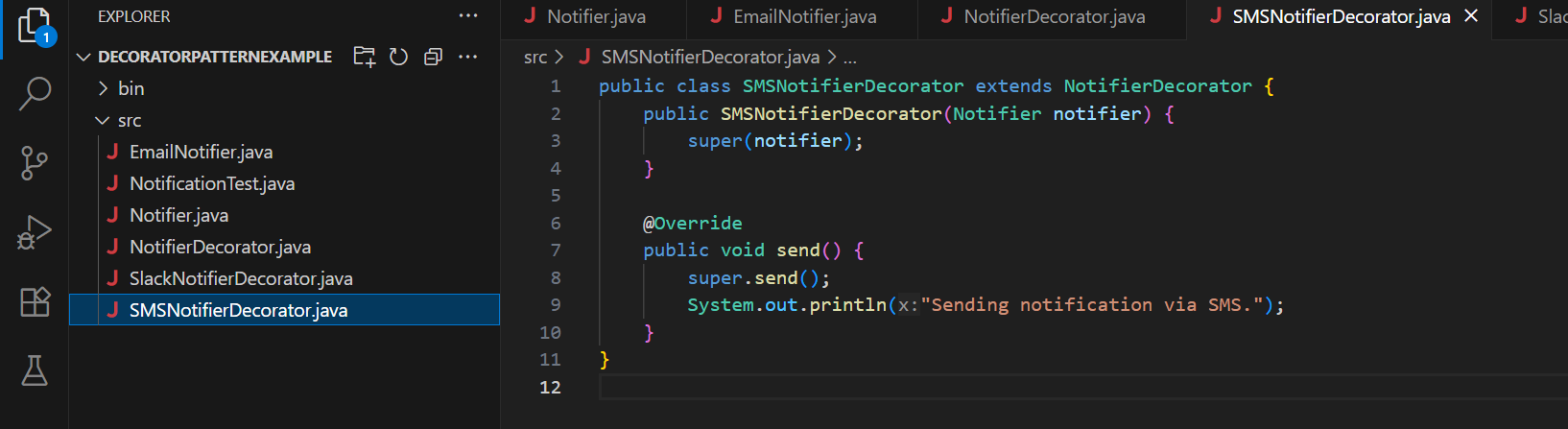
public void send() {

super.send();

System.out.println("Sending notification via SMS.");

}

}



5) SlackNotifierDecorator.java

public class SlackNotifierDecorator extends NotifierDecorator {

public SlackNotifierDecorator(Notifier notifier) {

super(notifier);

}

@Override

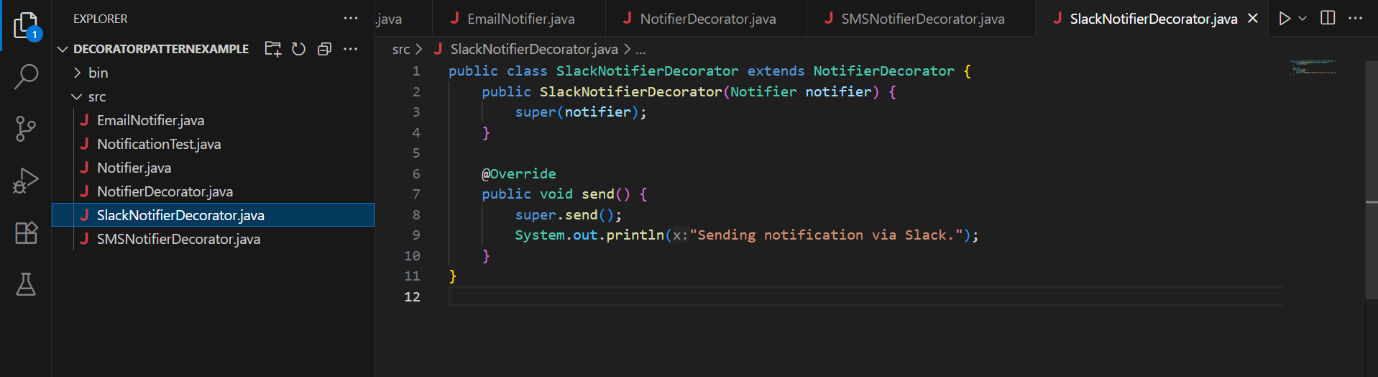
public void send() {

super.send();

System.out.println("Sending notification via Slack.");

}

}



6) NotificationTest.java

public class NotificationTest {

public static void main(String[] args)

Notifier emailNotifier = new EmailNotifier();

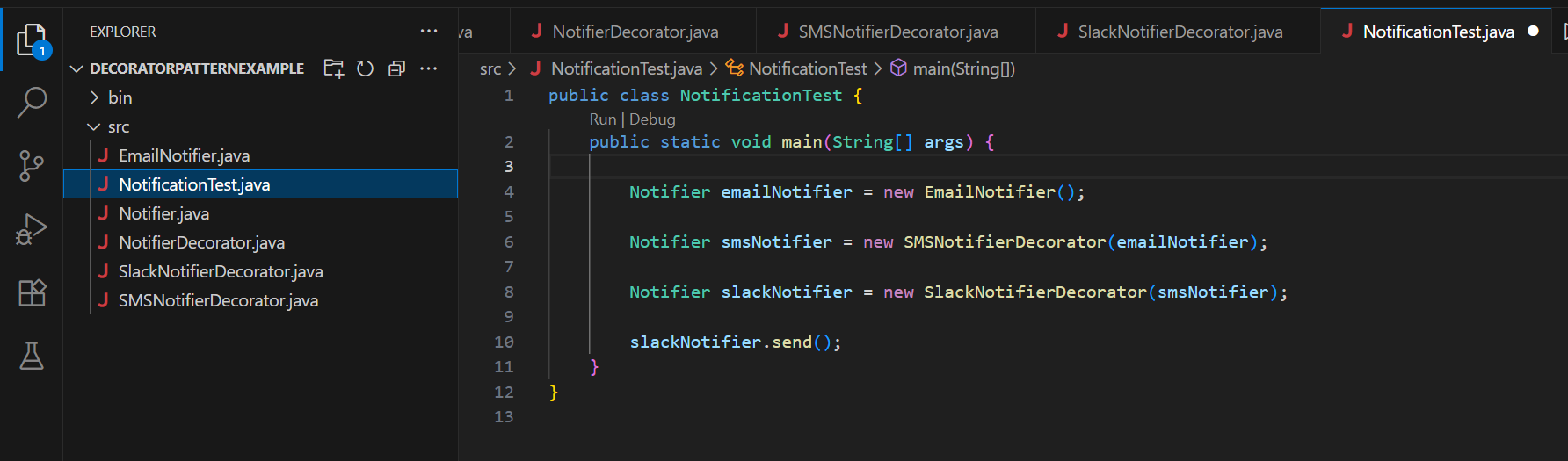
Notifier smsNotifier = new SMSNotifierDecorator(emailNotifier);

Notifier slackNotifier = new SlackNotifierDecorator(smsNotifier);

slackNotifier.send();

}

}



Output:

