Design patterns and principles

SuperSet ID:6412063

Exercise 9: Implementing the Command Pattern

Code:

public interface Command {

void execute();

}

public class LightOnCommand implements Command {

private Light light;

public LightOnCommand(Light light) {

this.light = light;

}

public void execute() {

light.turnOn();

}

}

public class LightOffCommand implements Command {

private Light light;

public LightOffCommand(Light light) {

this.light = light;

}

public void execute() {

light.turnOff();

}

}

public class RemoteControl {

private Command command;

public void setCommand(Command command) {

this.command = command;

}

public void pressButton() {

if (command != null) {

command.execute();

} else {

System.out.println("No command set.");

}

}

}

public class Light {

public void turnOn() {

System.out.println("The light is ON.");

}

public void turnOff() {

System.out.println("The light is OFF.");

}

}

public class HomeAutomationTest {

public static void main(String[] args) {

// Receiver

Light livingRoomLight = new Light();

Command lightOn = new LightOnCommand(livingRoomLight);

Command lightOff = new LightOffCommand(livingRoomLight);

RemoteControl remote = new RemoteControl();

remote.setCommand(lightOn);

remote.pressButton();

remote.setCommand(lightOff);

remote.pressButton();

}

}

Output:

A close-up of a text

Description automatically generated