Exercise 9: Implementing the Command Pattern

Objective

To build a home automation system using the Command Pattern, where commands to control devices (like turning lights on/off) are encapsulated as objects.

Java Code – CommandPatternExample.java

// Step 1: Command Interface interface Command {

void execute();

}

// Step 2: Receiver Class class Light {

public void turnOn() { System.out.println("The light is ON");

}

public void turnOff() { System.out.println("The light is OFF");

}

}

// Step 3a: Concrete Command - Light ON class LightOnCommand implements Command {

private Light light;

public LightOnCommand(Light light) { this.light = light;

}

@Override

public void execute() { light.turnOn();

}

}

// Step 3b: Concrete Command - Light OFF class LightOffCommand implements Command {

private Light light;

public LightOffCommand(Light light) { this.light = light;

}

@Override

public void execute() { light.turnOff();

}

}

// Step 4: Invoker 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 assigned to button.");

}

}

}

// Step 5: Test Class

public class CommandPatternExample {

public static void main(String[] args) { Light livingRoomLight = new Light();

Command lightOn = new LightOnCommand(livingRoomLight); Command lightOff = new LightOffCommand(livingRoomLight);

RemoteControl remote = new RemoteControl();

System.out.println("--- Press ON button ---"); remote.setCommand(lightOn); remote.pressButton();

System.out.println("--- Press OFF button ---"); remote.setCommand(lightOff); remote.pressButton();

}

}

Sample Output

--- Press ON button --- The light is ON

--- Press OFF button --- The light is OFF

Key Benefits of Command Pattern

Feature Description

Encapsulation Encapsulates actions/requests as command objects Loose Coupling Invoker and Receiver are decoupled

Command Queue Support Easy to implement undo/redo or delayed execution features

Reusable Commands Commands can be reused with different receivers