**Exercise 9: Implementing the Command Pattern**

**Scenario:** You are developing a home automation system where commands can be issued to turn devices on or off. Use the Command Pattern to achieve this.

**Steps:**

1. **Create a New Java Project:**
   * Create a new Java project named **CommandPatternExample**.
2. **Define Command Interface:**
   * Create an interface Command with a method **execute()**.
3. **Implement Concrete Commands:**
   * Create classes **LightOnCommand**, **LightOffCommand** that implement Command.
4. **Implement Invoker Class:**
   * Create a class **RemoteControl** that holds a reference to a Command and a method to execute the command.
5. **Implement Receiver Class:**
   * Create a class **Light** with methods to turn on and off.
6. **Test the Command Implementation:**
   * Create a test class to demonstrate issuing commands using the **RemoteControl**.

**Solution:**

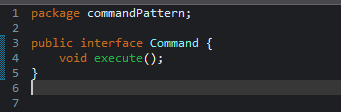
the Command Pattern to encapsulate requests like turning a Light ON/OFF as command objects.  
This lets you decouple the invoker (RemoteControl) from the actual operation logic (Light).

| **Component** | **Role** | **Purpose** |
| --- | --- | --- |
| **Command** | Command Interface | Declares method execute() |
| **LightOnCommand, LightOffCommand** | Concrete Commands | Implements command-specific action |
| **RemoteControl** | Invoker | Triggers command execution without knowing details |
| **Light** | Receiver | Contains the actual business logic (turn on/off) |
| **CommandTest** | Client | Sets and executes commands using RemoteControl |

**Code:**

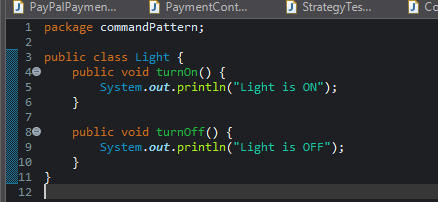
**Command.java**

* Declares a single method execute(), which all commands will implement.

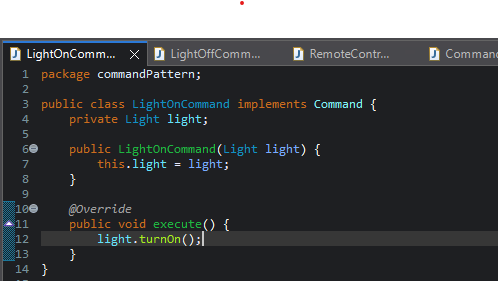


**Light.java**

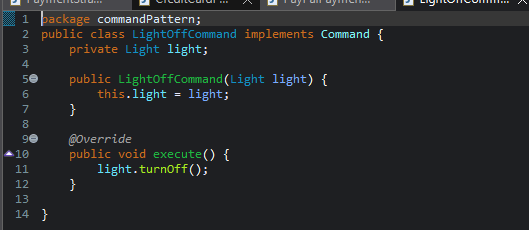
* Contains actual actions that will be performed.
* The command classes will call these methods.



**Concrete Commands:**

1. **LightOnCommand.java**

**b. LightOffCommand.java**

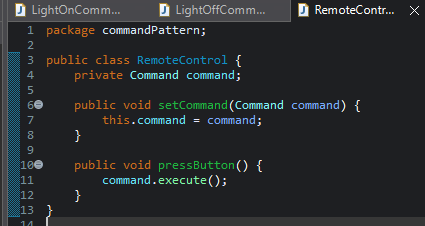
****

**Explanation:**

* Each command class holds a reference to the Light object (receiver).
* Implements the execute() method by calling turnOn() or turnOff().

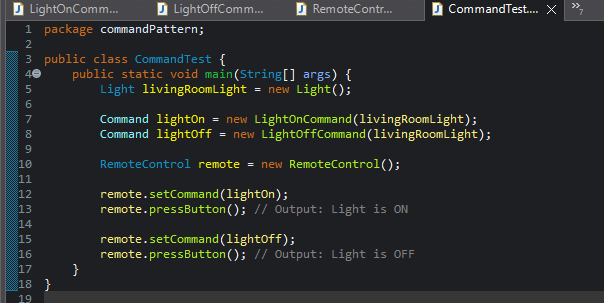
**RemoteControl.java**

* This is the invoker. It doesn’t know the details of the command; it just calls execute().

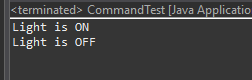
****

**CommandTest.java**

* Creates commands and sets them in the RemoteControl.
* When you press the button, the associated command is executed.

****

**Output:**

****