Object Oriented Programming OOP

Lab Project

Smart Home Controller

OOP-SE-Semester Project

Name 1: Muhammad Umar (L1F23BSSE0137)

Name 2: Duad Imran Bajwa (L1F23BSSE0066)

Section: O8

Instructor: Sir Fitrus Abbas

Date: 28-Jan-2025

Design Choices:

1. **Principles:**

    a. **Encapsulation**: Device properties (name, is On, brightness, temperature) are encapsulated within their respective classes, accessible only through getter and setter methods. This protects the data from unintended modification.

    b. **Inheritance**: A common SmartDevice class defines shared properties (name, on/off state). Specific devices like Light and Thermostat inherit this structure, extending it with device-specific attributes.

    c. **Abstraction**: SmartDevice is an abstract class that defines the blueprint for all devices, enforcing implementation of the getStatus() method in child classes.

    d. **Polymorphism**: Devices are handled as SmartDevice objects, allowing the program to manage different types of devices (lights, thermostats) uniformly.

2. **Graphical User Interface (GUI):**

    a. **Java Swing**: Chosen for simplicity and ease of use in creating desktop applications.

    b. **Modular Design**: The GUI is split into panels for specific functions:
        i. The main panel hosts the list of devices.
        ii. The button panel contains controls for adding, removing, and managing devices.

    c. **Dynamic Updates**: The GUI updates the device list and statuses automatically based on user actions.

3. **User Experience:**

    a. **Simple Navigation**: Users interact with the application through clear buttons and a list of devices.

    b. **Error Messages**: Basic error handling provides feedback for invalid actions (e.g., selecting no device to remove).

    c. **Interactive Controls**: Device-specific settings (brightness, temperature) are managed using sliders for a smooth experience.

4. **Scalability:**

    a. The program can easily extend to handle new device types by creating new classes inheriting from SmartDevice.

b. Future enhancements like saving and loading device states can be added with minimal changes to the core structure.

Class Structures:

1. Smart Device (Abstract Class)

- **Attributes:**

  name: The name of the device.
  isOn: The on/off state of the device.

- **Methods:**

  toggle(): Toggles the device's on/off state.
  getStatus(): Abstract method implemented by child classes to return the device's status.

2. Light (Inherits Smart Device)

- **Additional Attribute:**

  brightness: Integer value (0-100%) representing the brightness of the light.

- **Additional Methods:**

  getBrightness() and setBrightness(int brightness): Accessor and mutator for brightness.
  getStatus(): Returns the current state of the light (e.g., on/off, brightness).

3. Thermostat (Inherits Smart Device)

- **Additional Attribute:**

  temperature: Integer value representing the thermostat's temperature (15°C - 30°C).

- **Additional Methods:**

  getTemperature() and setTemperature(int temperature): Accessor and mutator for temperature.
  getStatus(): Returns the current state of the thermostat (e.g., on/off, temperature).

4. SmartHomeController (Main Class)

- **Attributes:**

  frame: The main application window.

mainPanel: The container for all GUI components.
deviceListModel: Model for the list of devices displayed in the GUI.
devices: Array List of all added SmartDevice objects.

- **Key Methods:**

  AddDevice (SmartDevice device): Adds a new device to the list.
  removeDevice(int index): Removes a device from the list based on the index.
  controlDevice(int index): Opens a dialog for managing the selected device's settings.

How to Use the Application:

1. Launching the Application

Run the SmartHomeController class.
A GUI window will open with an empty list of devices and buttons for actions.

2. Adding Devices

Click **"Add Light"** to add a light device to the list.
Click **"Add Thermostat"** to add a thermostat device to the list.
Each added device will appear in the list with its default settings.

3. Controlling Devices

- Select a device from the list and click **"Control Device"**.

  For **Light**:

  Adjust the brightness using the slider.
  Toggle the on/off state using the checkbox.

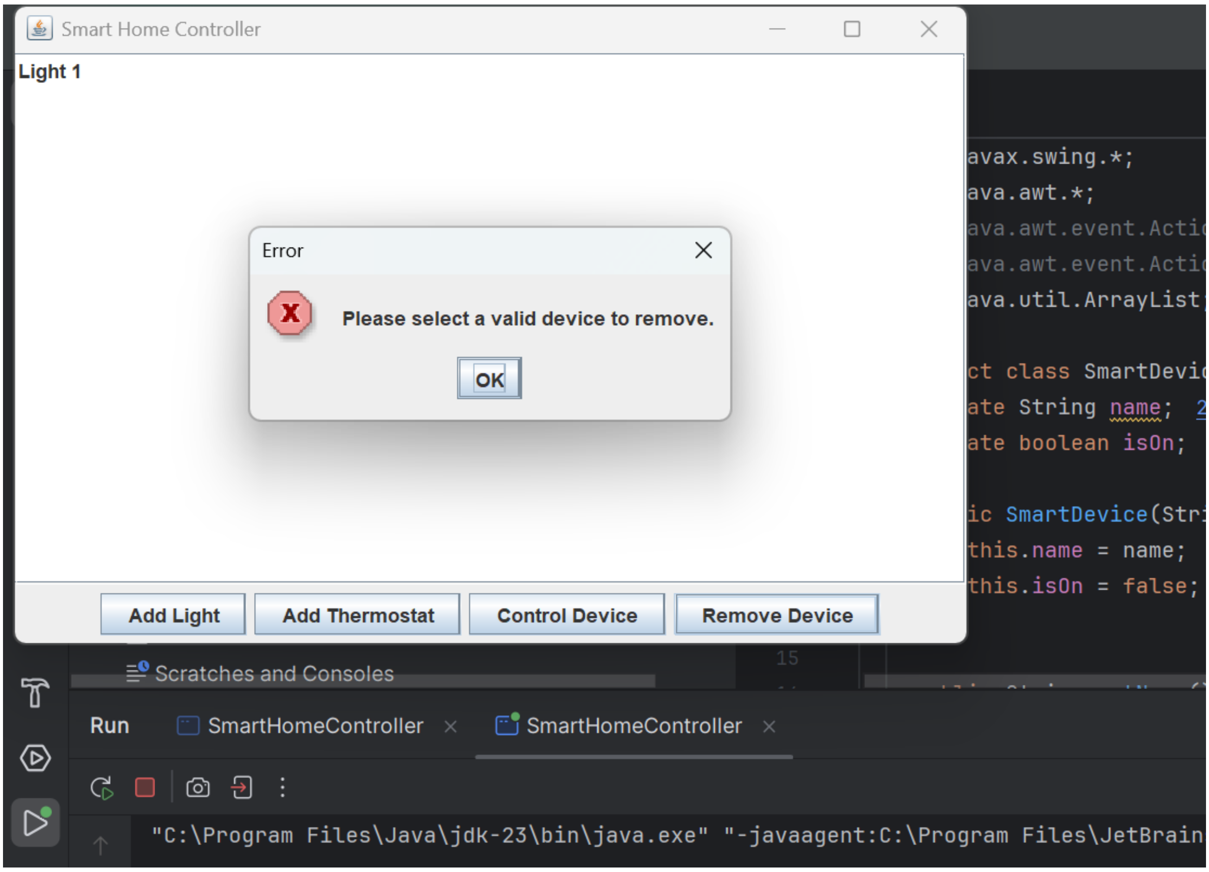
  For **Thermostat**:

  Adjust the temperature using the slider.
  Toggle the on/off state using the checkbox.

The changes will immediately reflect in the device's status shown in the list.

4. Removing Devices

Select a device from the list and click **Remove Device** to delete it.If no device is selected, an error message will appear.



Please select a valid device to remove.

```
avax.swing.*;
ava.awt.*;
ava.awt.event.Acti
ava.awt.event.Acti
ava.util.ArrayList

ct class SmartDevic
ate String name;  2
ate boolean isOn;

ic SmartDevice(Str
this.name = name;
this.isOn = false;
```

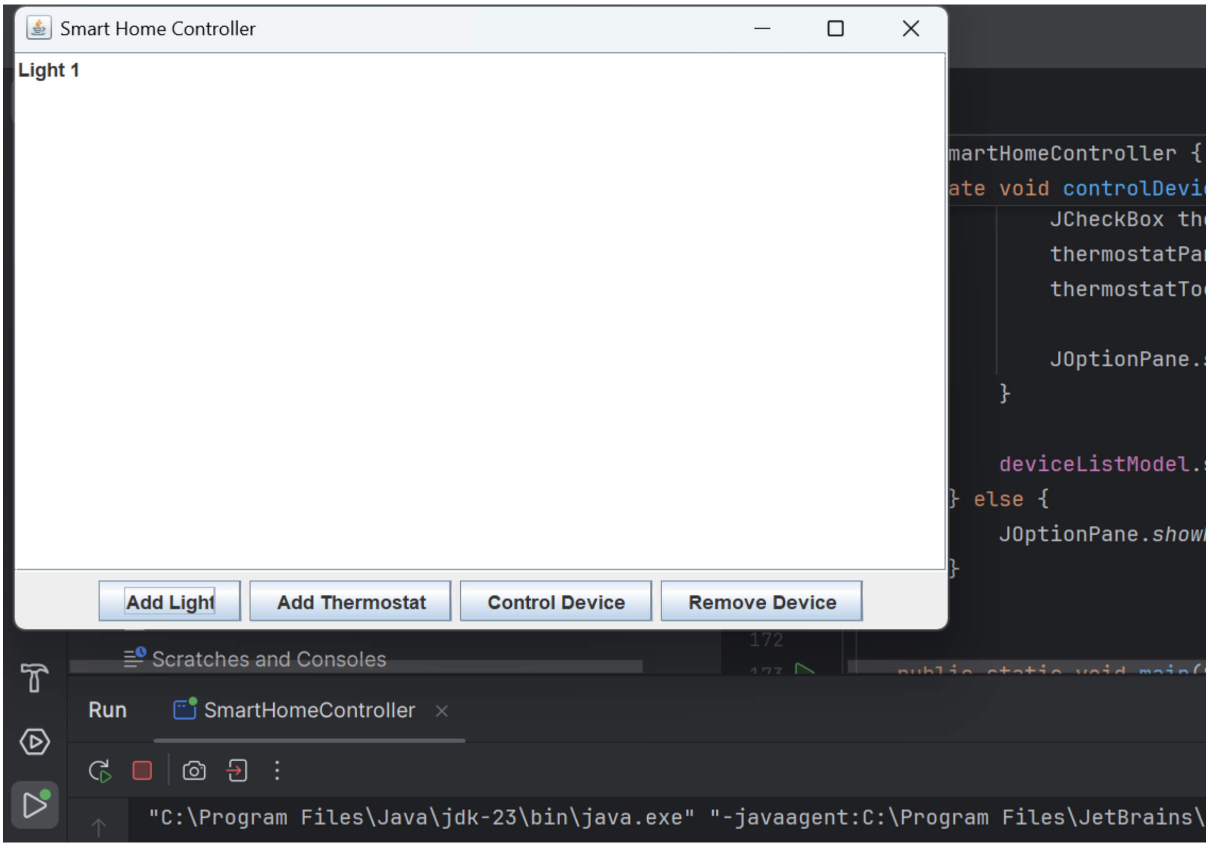
```
"C:\Program Files\Java\jdk-23\bin\java.exe" "-javaagent:C:\Program Files\JetBrain
```

5. Viewing Device Status

The device list dynamically updates to show the status of each device (e.g., on/off state, brightness, temperature).

Example:

Click **Add Light**.

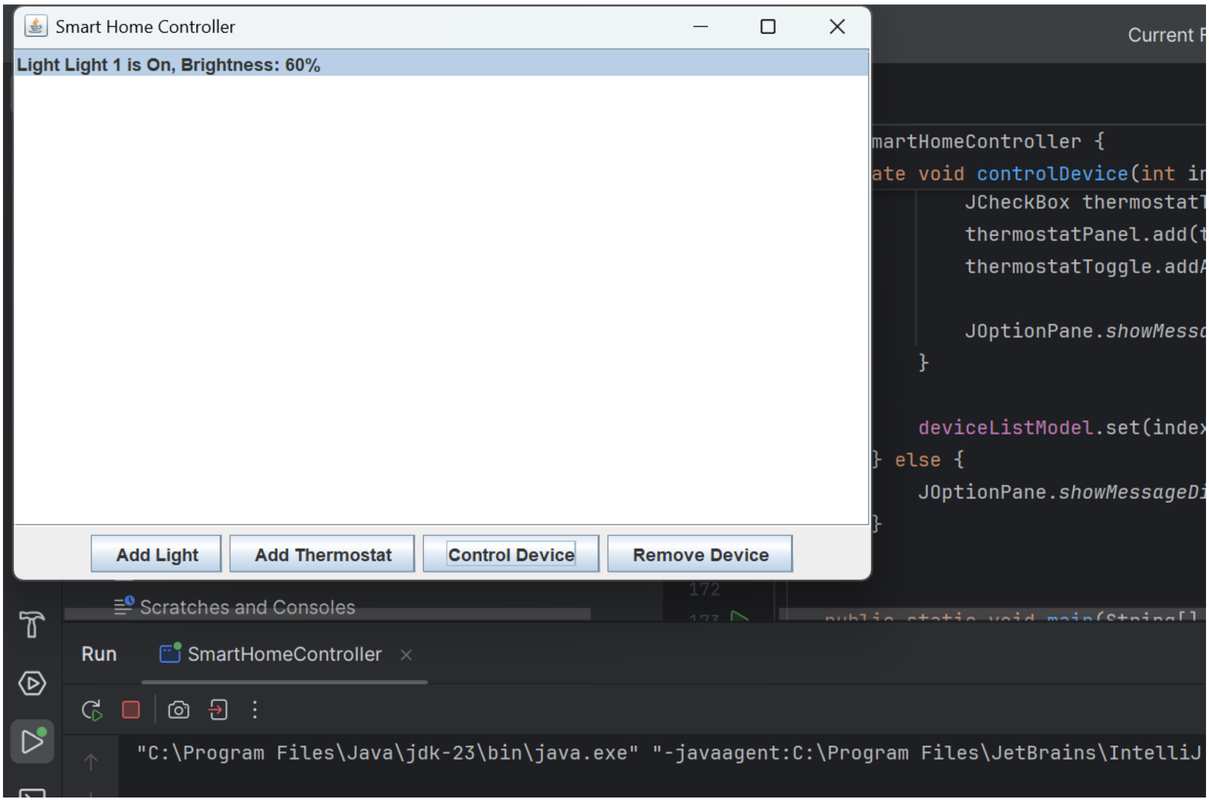


Light 1

```
martHomeController {
ate void controlDevi
        JCheckBox th
        thermostatPa
        thermostatTo

        JOptionPane.
    }

    deviceListModel.
} else {
    JOptionPane.show
}

public static void main(
```

```
"C:\Program Files\Java\jdk-23\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\
```

Select the light, click **Control Device,** and adjust brightness to 60%. Toggle it on:

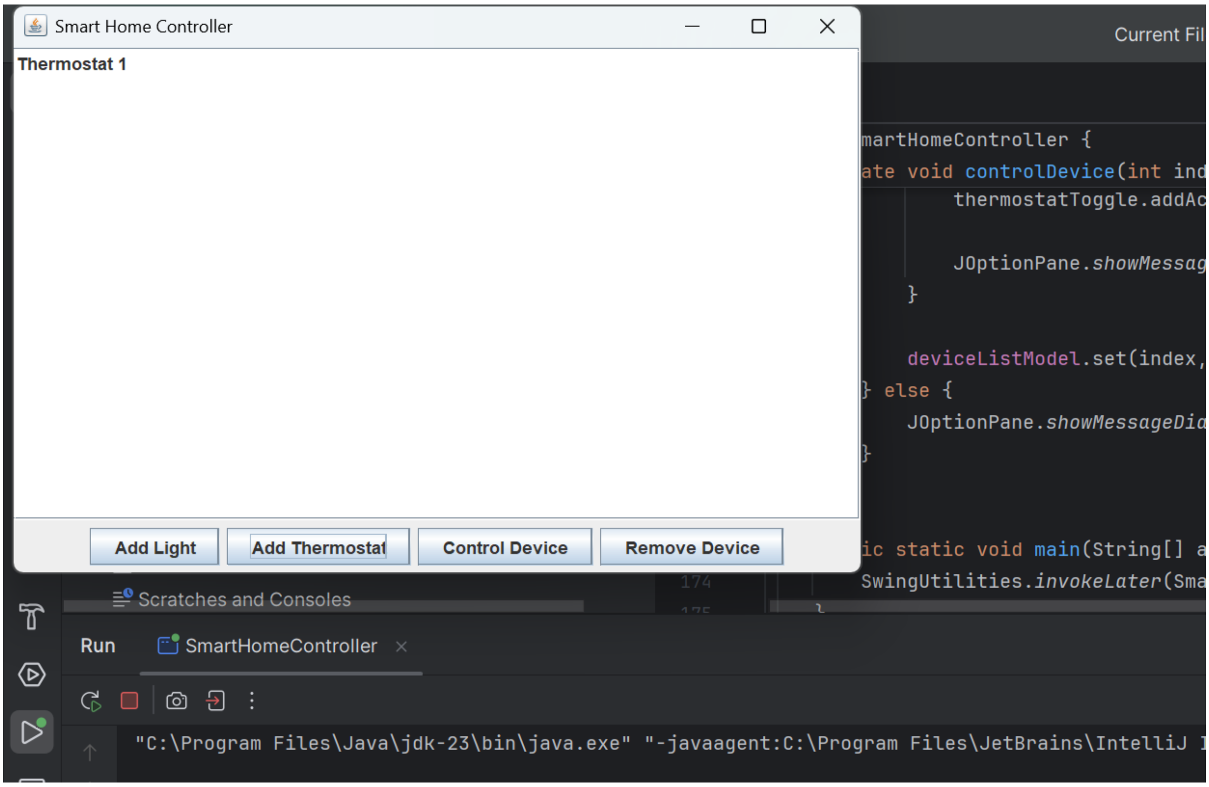


Light Light 1 is On, Brightness: 60%

```
martHomeController {
ate void controlDevice(int in
        JCheckBox thermostatT
        thermostatPanel.add(t
        thermostatToggle.addA

        JOptionPane.showMessa
    }

    deviceListModel.set(index
} else {
    JOptionPane.showMessageDi
}

public static void main(String[]
```

```
"C:\Program Files\Java\jdk-23\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ
```

Click **"Add Thermostat"**

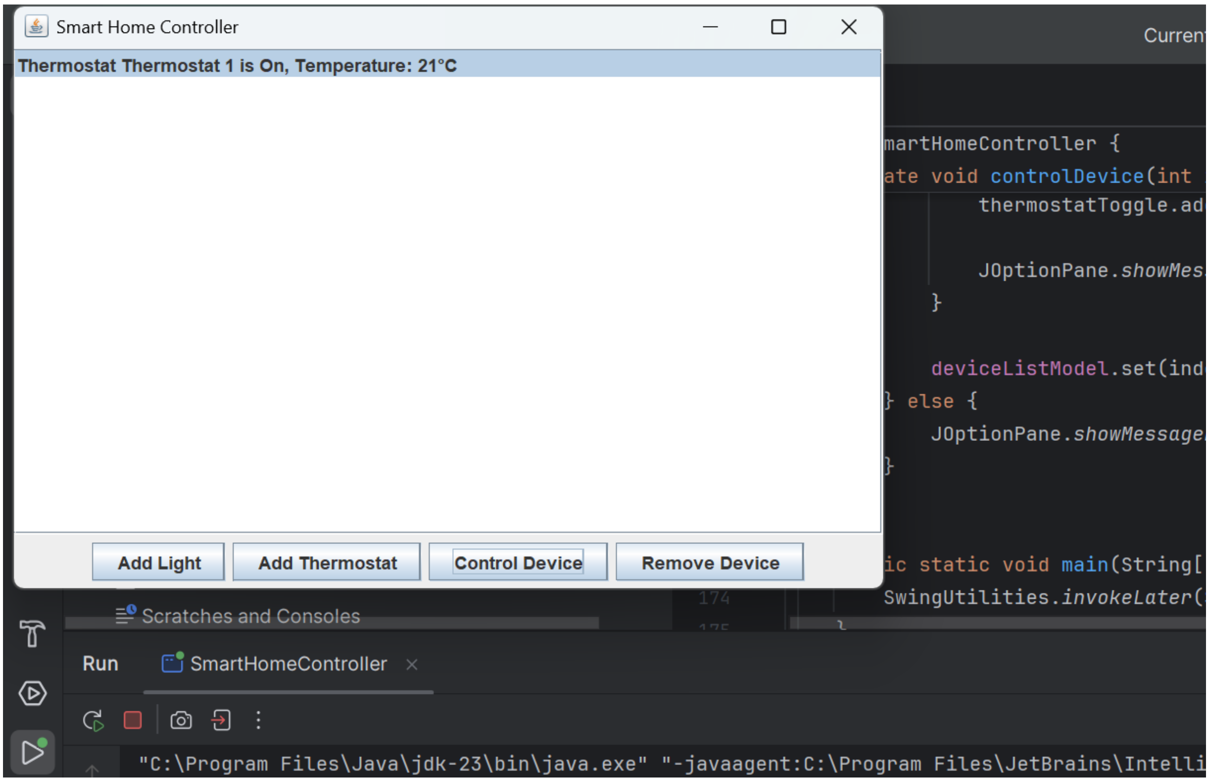


```
Thermostat 1
```

```
martHomeController {
ate void controlDevice(int ind
        thermostatToggle.addAc

        JOptionPane.showMessag
    }

    deviceListModel.set(index,
} else {
    JOptionPane.showMessageDia
}

ic static void main(String[] a
SwingUtilities.invokeLater(Sma
```

```
"C:\Program Files\Java\jdk-23\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ I
```

Select the thermostat, click **Control Device,** set temperature to 21°C, and toggle it on:



```
Thermostat Thermostat 1 is On, Temperature: 21°C
```

```
martHomeController {
ate void controlDevice(int
        thermostatToggle.ad

        JOptionPane.showMes
    }

    deviceListModel.set(ind
} else {
    JOptionPane.showMessage
}

ic static void main(String[
SwingUtilities.invokeLater(
```

```
"C:\Program Files\Java\jdk-23\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\Intelli
```