**Smart Device Management Application**

**Overview**

This application is a GUI-based Smart Device Management System that allows users to add, configure, and remove various smart devices such as Light Bulbs, Fans, Thermostats, and Cameras. The application is designed using Java Swing for the graphical user interface and employs object-oriented programming principles for modularity and extensibility.

**Class Structures**

1. **Abstract Class:**

**Smart Device**

* **Purpose**: This serves as the base class for all devices. It holds common properties and methods for the devices.
* **Attributes:** String name, Boolean state
* **Methods:** get Name (), set Name (), is State (), set State ()
* Serves as a blueprint for all device types.

1. **Concrete Classes:**These classes extend the Smart Device class and represent individual smart devices.

**Lightbulb**:

* **Purpose**: Represents a light bulb smart device.
* **Attributes:** int brightness, String color
* **Additional Methods:** get Brightness (), set Brightness (), get Color (), set Color ()
* **Constructor:** Accepts name and default color.

**Fan**:

* **Purpose**: Represents a fan smart device.
* **Attribute:** int speed
* **Additional Methods:** get Speed (), set Speed ()
* **Constructor:** Accepts name and default speed.

**Thermostats**:

* **Purpose**: Represents a thermostat smart device.
* **Attribute:** double temperature
* **Additional Methods:** get Temperature (), set Temperature ()
* **Constructor:** Accepts name and default temperature.

**Cameras**:

* **Purpose**: Represents a camera smart device.
* **Attributes:** String resolution, String format
* **Additional Methods:** get Resolution (), set Resolution (), get Format (), set Format ()
* **Constructor:** Accepts name, resolution, and format.

1. **Main Class: Project:**

* **Purpose:** Entry point for the application.
* Initializes the App class.

1. **App Class**

* **Purpose**: This class represents the graphical user interface (GUI) for the application.
* **Fields**:
  + APP\_WIDTH, APP\_HEIGHT: The dimensions of the app's window.
  + outer Frame: The main window of the app.
  + hidden Window: A panel that holds other components, which is used for the device interaction (adding, configuring, and removing devices).
  + buttons: An array of buttons that users can interact with (Add Device, Configure Device, etc.).
  + devices: A list of device names for selection.
  + added Devices: A list that holds the added smart devices.
* **Constructor**:
  + Initializes the frame and sets up the basic GUI layout.
  + background Image: Sets the background image.
  + outer Frame: Creates a JFrame window with specified size and location.
  + buttons: Creates five buttons for different actions (Add Device, Configure Device, etc.).
  + The GUI components are laid out in a JPanel with custom painting to show the background and other components.
* **GUI Layout**:
  + The JPanel for the main window is customized to draw the background image and apply transparency on the left side.
  + The JPanel containing the buttons is placed on the right side of the window with a Box Layout.
  + Each button is added to the panel and listens for user clicks through the Button Listener.

1. **Button Listener Class:**

* **Purpose**: This is the action listener for handling button clicks.
* **Add Device**:
  + A dialog box allows the user to choose a device (Lightbulb, Fan, Thermostat, Camera).
  + After selecting the device, the user is prompted to enter a name for the device.
  + The new device is created and added to the added Devices list.
* **Configure Device**:
  + If there are devices in added Devices, the user can choose a device to configure.
  + For each device type (Lightbulb, Fan, Thermostat, Camera), different configuration options are shown (e.g., brightness for Lightbulb, speed for Fan).
  + After adjusting the settings, the configuration is applied to the device.
* **Remove Device**:
  + The user can remove a device from the list of added devices.
  + A list of the added devices is shown, and the user can select and remove one.
* **Credits**:
  + Displays a "Credits" window with acknowledgments for the developers and contributors.

1. **Event Handling:**

* Each button has a corresponding action handler to perform the appropriate task.
* For example, when the "Add Device" button is clicked, a dialog appears allowing the user to select a device and assign it a name.
* When the "Configure Device" button is clicked, the user is presented with a list of added devices to configure their properties.

1. **Panels and Dialogs:**

* The J Option Panel is used to show dialogs for user interactions, like adding or configuring a device.
* Custom panels are created for each configuration screen, depending on the selected device type.

1. **Swing GUI Components:**

* **JButton**: Used for interactive buttons like "Add Device", "Configure Device", etc.
* **JPanel**: Used for layout and custom components like device configuration.
* **JList**: Used to display the list of devices for adding or removing.
* **JSlider**: Used for adjusting properties like brightness, speed, temperature.

**Summary of Flow:**

1. The app opens with a main window that has buttons for interacting with devices.
2. The user can add a new device, configure an existing one, or remove a device.
3. The devices' properties can be adjusted based on the device type.
4. The added devices are stored in an Array List and can be accessed for configuration or removal.
5. The app provides an intuitive interface for managing these smart devices.

**Design Choices:**

* **Abstract Class for Common Behavior**: The Smart Device class serves as a parent class for different device types. This allows you to extend it and add specific properties and methods for each device.
* **Use of JPanel and JOptionPanel for GUI**: These are standard Swing components for creating interactive and visually appealing interfaces.
* **ActionListener**: A centralized Button Listener class handles all button actions, making it easier to manage and expand functionality.
* **Customization and Extensibility**: Adding more devices or features can be done easily by adding new classes that extend Smart Device and adding corresponding GUI components in the Button Listener class.  
    
  **How to Use the Application**

1. **Starting the Application**:
   * Run the Project class.
   * The main window will appear with buttons for various actions on the left-hand side.
2. **Adding a Device**:
   * Click the "Add Device" button.
   * Select the device type (Lightbulb, Fan, Thermostat, or Camera) from the displayed list.
   * Enter a unique name for the device.
   * The device will be added to the system and displayed in the list of configured devices.
3. **Configuring a Device**:
   * Click the "Configure Device" button.
   * Select a device from the list of added devices.
   * Modify the attributes specific to the device type (e.g., brightness and color for Lightbulb).
   * Confirm the changes to update the device.
4. **Removing a Device**:
   * Click the "Remove Device" button.
   * Select the device to remove from the list of added devices.
   * Confirm the removal.
5. **Exiting the Application**:

Click the "Exit" button to close the application

**Conclusion:**

This application provides a foundation for managing smart devices in a modular and scalable way. Its intuitive interface and flexible design make it a starting point for more advanced smart home management solutions.