HACETTEPE UNIVERSITY

COMPUTER ENGINEERING DEPARTMENT

BBM104 **2023 SPRING**

Assignment 2

April 21, 2023

Student Name	*Student Number*
Yusuf İpek	2220356048

Index

Problem

“Humankind is a lazy thing indeed, according to our evolution progress, we always tried to do something with less and less effort. So, we invented many things to make our life easier, so that we can have more time to do "nothing".” One of them is smart home system.

So, the main purpose of this project assignment is developing a smart home system with different smart devices such as Smart Lamp, Smart Color Lamp, Smart Plug and Smart Camera.

The system will receive commands with text files such as adding or removing devices, setting device properties (kelvin, ampere, mb, etc.), and time settings. These commands will be executed with proper error handling.

The aim is to make daily life easier for users by developing a maintainable and efficient smart home system using OOP design principles and Java programming language.

Solution Approach

I divided the problem to small pieces as Input-Output Handling, Devices Classes, Controllers.

- The “Input-Output Handling” part includes the reading input file, output strings which is reachable everywhere (with static keyword) and writing output file to this output strings.
- The “Devices Classes” part includes finding common properties of all devices.
    - Writing the base abstract class for all devices “SmartDevice” with common properties.
    - Writing the devices own classes as sub class of “SmartDevice”.
        - Color lamps can be considered a subclass of the lamp class due to their shared properties such as brightness and kelvin.
    - Overriding the methods (like toString) of each device and writing new methods for their specific properties (calculate MB, calculate watt etc.).
- The Controllers part includes 3 controller class.
    - There is a time controller class for the managing time.
        - Setting time,
        - Keeping the time of now,
        - Skipping minutes.
        - Padding time if it is not padded yet (formatting)
        - Parsing the time to string and string to time (using simple date formatter).
    - Item Controller is for managing items.
        - Adding- removing items to home system,
        - Keeping added items in device list,
        - Sorting items by their switch time and natural order.
        - Changing properties of items (status, name etc.)
        - Some checkers for validation (check name, check status etc.).
    - The last of these classes is command controller class which has the other controller classes as parameters. The main purpose of this class is parsing commands and calling related function from item controller or time controller.

Some Problems and Solutions

I faced several problems during the project assignment and solve them by making online research, asking chatGPT etc. Some of these problems are working with date object, sorting items and there is a lot of work that needs to be done.

- Working with date objects problem is about parsing and formatting the time strings to “Date” object and padding the time strings.
    - My solution is making online research. At the end, I used the simple date format and wrote a function for padding, checking the length of the time and add “0” if it is necessary.
- Sorting items: Sorting items by switchtime which is nullable variable is hard for me, and sorting items which has the same switch time is a bit harder when they switch together.
    - My solution is learning collection sort, comparing by value and natural order by online research.
    - For switching items, I sort the list every switching but for the same switch time devices I wait to change the device’s switch time value for sorting, and bingo I solved it.
- There is lots of case, which is waiting to handle such as power calculation, storage calculation, errors (a lots of errors), setting time, a lots of properties for devices etc.
    - I figure out these issues by breaking down the project into smaller parts and focusing on one at a time, rather than trying to write all of them simultaneously.

Benefits of This System

The users can control the smart devices easily by this system. Also, the system keeps the devices list and shows the information such as name, device type, status, switch time and some special properties. So, managing items are easier and safer thanks to this smart home system.

Benefits of OOP

OOP (Object-Oriented Programming) has a lot of benefits that make it a popular programming paradigm. Reusing code and modularity are key benefits of OOP, which can help to development and minimize the risk of errors.
OOP also allows maintainability and scalability of code, in changes or additions can be made to individual objects or classes without affecting the entire program (for dividing small pieces in solution approach).
So, OOP is useful paradigm for creating complex systems like smart home system.

Four Pillars of OOP

Encapsulation

It is for hiding data and methods from the outside world (with private keyword) and accessing them only through specific methods. Encapsulation allows developers to control access to data. Thanks to this pillar the devices check their specific values before setting them such as MB values, ampere values, kelvin values etc.

Abstraction

Abstraction is another pillar of OOP that allows developers to focus on the features of an object, while hiding unnecessary details. This pillar helps to increase code modularity, allowing developers to change the implementation of an object without affecting the rest of the system. The example of it is base class “SmartDevice”.

Inheritance

Inheritance is a main concept of OOP. Creating new class based on an existing class is possible thanks to this pillar. Inheritance helps to increase code reusability and reduce development time and effort. Also, method overriding is an important feature of inheritance that enables developers to modify the behavior of existing code without changing the original code. This enables to create more specialized classes for specific needs. You can see this pillar's example in sub classes of "SmartDevice" Plug, Camera, Lamp. Also, the color lamp is shared more properties with the lamp device than others, so color lamp class is inherited from the lamp class.

Polymorphism

Polymorphism is the fourth pillar of OOP which refers to the ability of objects to take on multiple forms. It means, polymorphism is using for represent multiple classes with single base class. This pillar is essential when building complex systems with a lot of classes that have similar properties but different implementations. Polymorphism helps to increase code flexibility, making it easier to add new features and update the system as requirements change. The example of this is device list of "SmartDevice" which includes plug, color lamp, lamp, camera devices. Using these classes methods without casting is possible thanks to this pillar.

UML Diagram and Explanation

UML Diagram

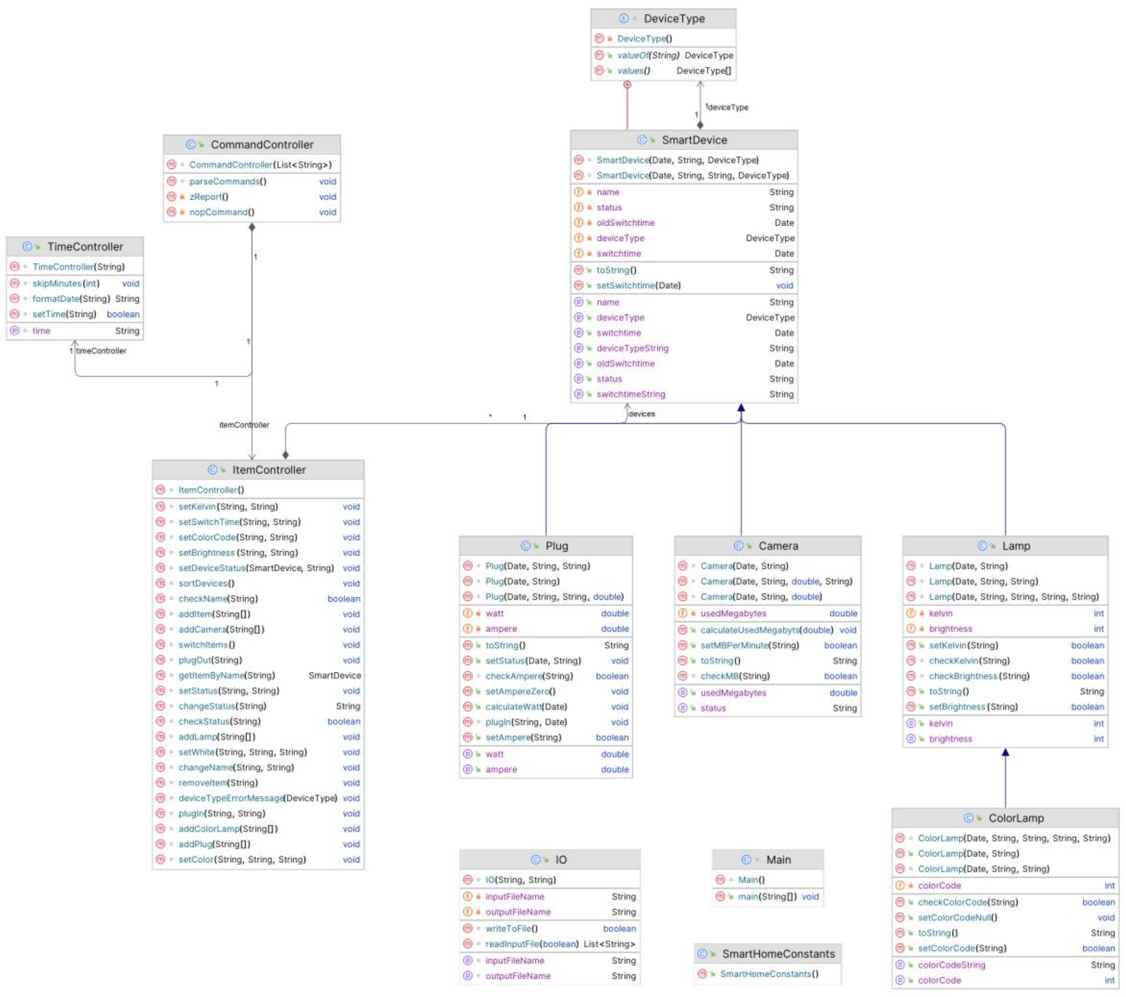

Image1 UML diagram of project assignment 2

Explanation

There are 11 classes as you can see in the picture. and additional details about some of these classes can be found in the Solution Approach section. The classes include:

- “SmartHomeConstants” which contains messages and commands as strings.
- “SmartDevice” an abstract base class of other devices such as Plug, Camera and Lamp.
- “ColorLamp” a subclass of “Lamp” class that extends from “SmartDevice”.
- “CommandController” class is connected the other controller classes for managing time and items.
- “IO” class handles input-output operations.

Resources

- FormattingDate-JavatPoint: https://www.javatpoint.com/java-string-to-date (Access Date: 15.04.2023)
- DateFormater–DigitalOcean: https://www.digitalocean.com/community/tutorials/java-simpledateformat-java-date-format (Access Date: 15.04.2023)
- JavadocComments-Oracle:https://www.oracle.com/technical-resources/articles/java/javadoc-tool.html (Access Date: 18.04.2023)
- StackOverFlow: https://stackoverflow.com/ (Access Date: 19.04.2023)
- OpenAI – ChatGPT: https://openai.com/blog/chatgpt (Access Date: 21.04.2023)
- UmlDiagram - JetBrains: https://www.jetbrains.com/help/idea/class-diagram.html (Access Date: 21.04.2023)