Hacettepe University

Department of Artificial Intelligence Engineering

BBM104 Assignment 2 Report

Bora Dere – 2220765021

21/04/2023

Problem

The problem is that mankind has grown lazier day by day and invented things to make their life easier, so that they can have more time to do “nothing”. Their deep desire to achieve full automation does not look like it is going to stop somewhere. Since this is the case, we help mankind by creating a functional smart home design, so they can keep doing “nothing”. This smart home design must be easily integratable to the actual life, it must be easy to use and understand, since easier life created a lazier generation.

Solution Approach

OOP (Object Oriented Programming) is the best option to use for this system. It helps us to split the program into pieces which makes it easier to understand, develop and change when needed. Dividing the code into classes for every smart device and interfaces for most of the operations keeps the whole code clean and tidy.

Problems Faced

Since OOP is a whole new thing, it was pretty difficult to find a way. This is a complex and detailed assignment, trying to find the best point to start was as crucial as coding another piece of the program. Different usages, unfamiliar type-castings etc. did not help either. Some problems faced:

1- Strange order of devices being printed: It is solved by using a special comparator which has been explained in the code.
2- Moving operations to interfaces caused currentTime not to be updated: Returning the needed value solved this problem.
3- Flexible format of time: Regex usage was helpful on the way of solving this problem

Benefits of This System

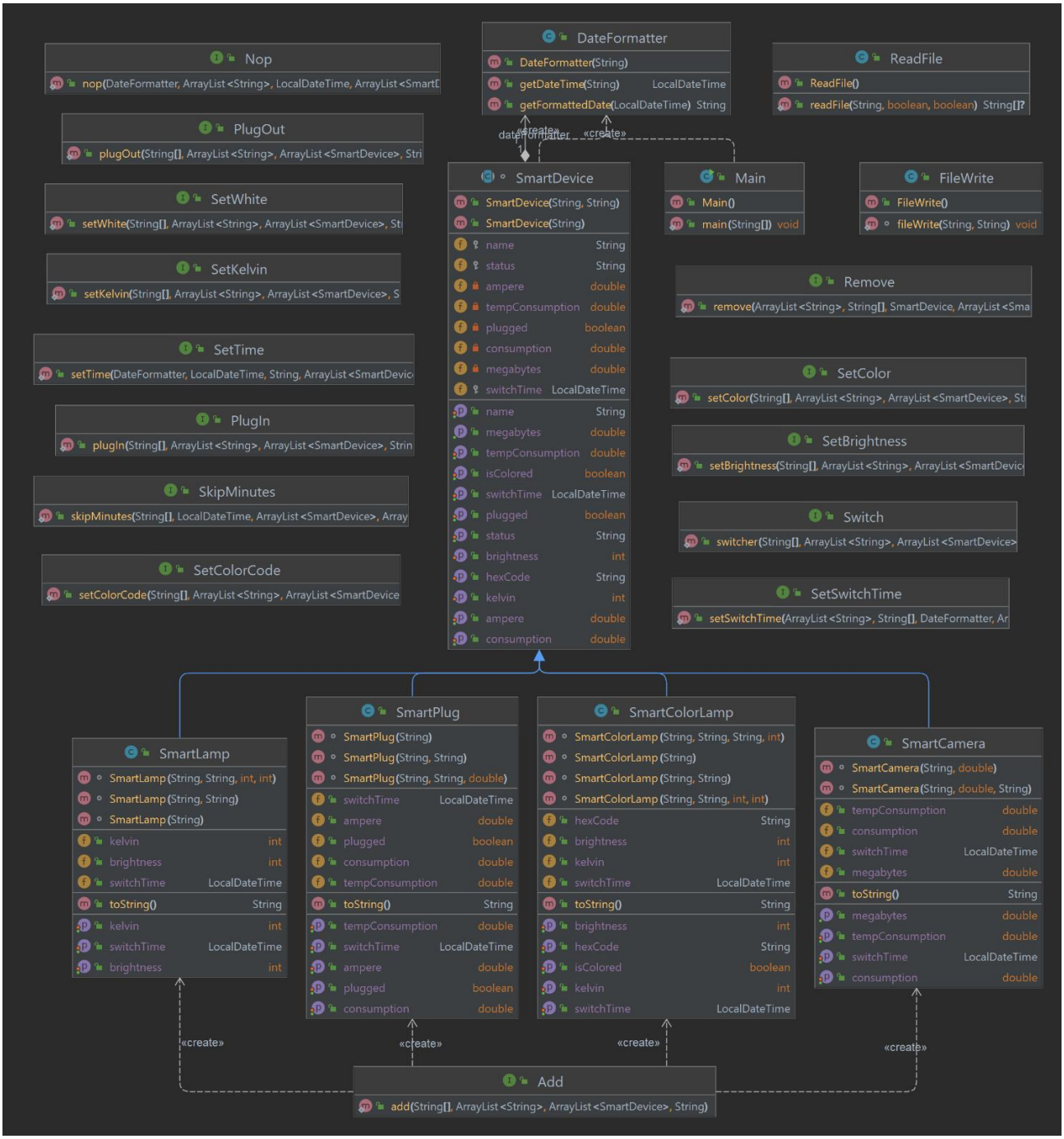
This system actually can be used in real scenarios too. It can be changed in a way that it works real-time and not on a user-switched-time basis. Since those systems are actually pretty common and requested, this is a good practice to be prepared.

Benefits of OOP

This code can be done in the traditional way too. But using OOP just gives the power of flexibility which increases imagination of the programmer. And that allows programmer to create a more rigid and beneficial system.

Pillars of OOP

1- Encapsulation: Packing data and methods in a class so if needed, their access can be arranged in a way to become inaccessible unless permitted.
2- Inheritance: While deriving a subclass from a superclass, that subclass inherits superclass' attributes and methods. This increases modularity.
3- Polymorphism: Objects being able to have different forms and/or behaviors. This increases flexibility of the code, makes it easier to use.
4- Abstraction: It gives programmer the ability to ignore irrelevant details of real-world objects and helps them to focus to the needed specifications.



As can be seen, SmartDevice is the superclass for smart devices. Add interface is directly related with 4 smart device classes. Other interfaces receive and update data from device classes.