ASSIGNMENT 2

Emre Çiğil

2210356033

b2210356033@cs.hacettepe.edu.tr

Problem Of Project

In this project, the operation of a smart home system has been requested in a programming language. For this system to be efficient, understandable, and easily revised, it must be written according to OOP rules. Multiple Java classes are needed to achieve this structure according to these rules. A Main Class, which manages and controls these classes, is required. The complexity of the system prolongs the coding process and makes creating UML diagrams more challenging. High concentration is necessary to consider all possible errors. Properties are assigned for each object created, and these properties must be securely stored, used, or modified when needed. These requirements make the project even more complex. To successfully complete this project, it is necessary to know Java methods and, especially, OOP rules.

Solutions Of Problem

To successfully solve these problems, it is essential not to dive directly into the code. A few days should be allocated for the thinking process. During this time, questions such as "what can be done, which path can be taken, how many different Classes can be created?" should be considered. Then, a UML diagram is drawn. The UML diagram alone is insufficient as a template for the project. In addition, notes should be taken on the paths to be followed, methods to be used in Classes, packages to be used, loops, try-catch blocks, and if-else conditions. Many errors need to be caught during the writing process. To achieve this, the code is frequently debugged and monitored. Sample inputs are used, and "successful" or "unsuccessful" messages are printed for some parts. There are multiple types of objects, each with its unique properties, as well as shared properties. Several Classes should be used to call, modify, or delete these objects as needed. By using a common class, other classes inherit from this class, eliminating code redundancy. This makes the code dynamic and allows for intervention when desired.

Problems Encountered

During the coding process, nested if-else patterns created significant difficulties in catching errors or creating new objects. To address this issue, Boolean variables were used, allowing for better control of the operations.

Another problem was establishing healthy communication between classes. If a class inherits from another class, or if subclasses are created from the main class using polymorphism, a UML diagram is drawn. Important notes are taken to avoid confusion in the subsequent process. Each class is used appropriately and at the right time.

Another issue is time management within the project. To solve this problem, the algorithm is considered, and notes are taken. Necessary packages are imported to facilitate operations related to date and time. With these packages, the time stored as a String can be easily formatted, allowing for seamless operations on time. For example, two times can be compared, and minutes can be added to the current time.

Benefits of OOP

Object-oriented programming (OOP) makes the software development process more comprehensible, flexible, and scalable. The primary benefit of OOP is that it allows us to design complex systems in a more organized and modular structure by using classes and objects that simulate real-world entities and concepts. This, in turn, increases code reusability and facilitates the expansion and maintenance of projects. OOP also ensures stronger and more secure applications through fundamental concepts such as abstraction, inheritance, polymorphism, and encapsulation. Abstraction simplifies the problem-solving process, making it more straightforward and adaptable, while inheritance allows for the reuse of code among classes that share common properties and functions. Polymorphism enables different objects to perform the same operation, providing a more generic and flexible code. Encapsulation protects the internal data and operations of classes, enhancing security and reliability.

Benefits of System

This system provides insights into the harmony and operation of devices in smart homes. The importance of OOP in the system becomes clear, and it serves as a valuable experience for using OOP in the future. It enables the use of packages within Java. Numerous commands and methods are learned.

OOP And UML Diagram

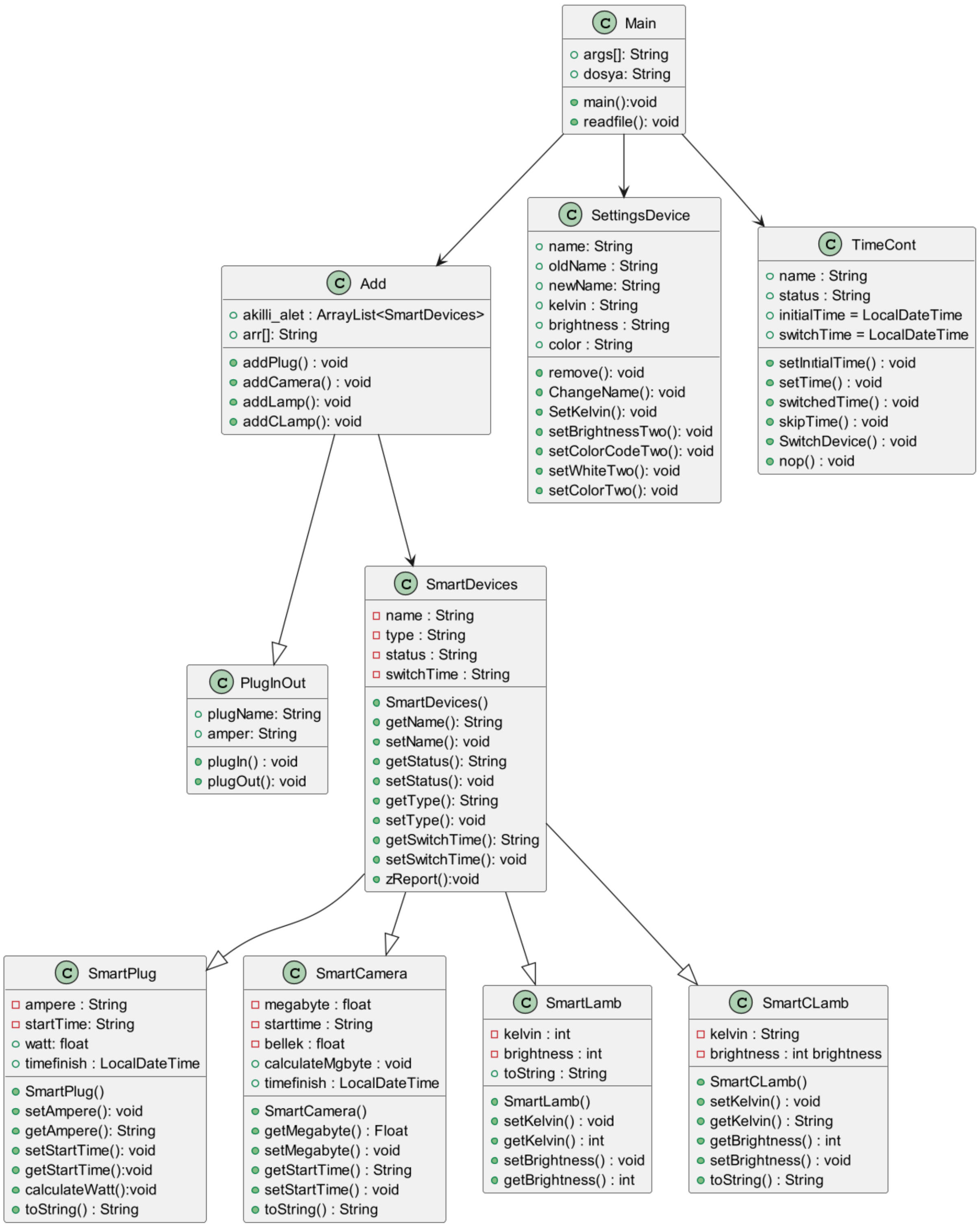
Abstraction: Abstraction aims to simplify real-world entities and concepts by breaking complex systems and processes into simpler components and presenting only relevant details to the programmer. This allows developers to independently handle parts of larger and more complex systems.

Encapsulation: Encapsulation conceals a class's internal structure and operations from the outside world, allowing interaction only through specific interfaces. This ensures the protection of the class's internal data and processes, thus increasing security and reliability. Encapsulation facilitates maintenance and updates by preventing changes to a class's internal structure from affecting external code.

Inheritance: Inheritance enables classes to share common properties and processes. This allows for the transfer and reuse of a class's properties and functions to another class. Inheritance increases code reusability and reduces unnecessary repetition, enabling us to write more organized and effective code.

Polymorphism: Polymorphism is the ability of different objects to perform the same operation. This allows for more general and flexible code. Polymorphism enables a method or process to be used by multiple objects or classes, thus increasing the code's overall applicability.

UML (Unified Modeling Language) is a standardized modeling language used for the visual representation of software systems. With an approach based on object-oriented programming, UML helps document and design the connections and structure among system classes, objects, relationships, and operations. Diagrams created with UML facilitate the understanding and sharing of software architectures and designs, making software development processes more efficient.



Resources

- Geekforgeeks.com
- Stackoverflow.com
- Open.ai
- youtube.com/@CodingWithJohn
- youtube.com/@YazlmBilimiAnkara
- Lecture PDF