

Design Patterns Used:

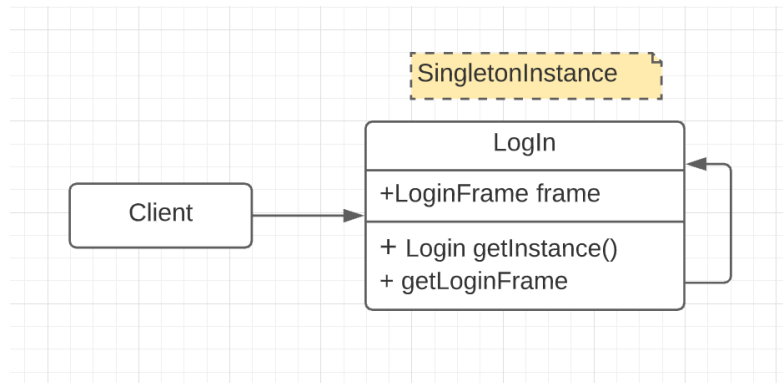
1. Singleton Design Pattern
 - Login Class
2. Factory Pattern
 - OperationGUI Factory Class
3. Builder Pattern
 - RecyclableItem Interface
4. Iterator Design Pattern
 - RCMIterator
5. Facade Design Pattern
 - StationUserInterface Class
6. Flyweight Design Pattern
 - RecyclingStation
 - RecyclingMachine
7. ObserverPattern
 - Publisher: Recycling Station
Subscriber: Customer Interface, StationUserInterface
 - Publisher: Recycling Machine
Subscriber: Customer Interface

The diagram illustrates several design patterns in UML:

- Singleton:** A `Client` interacts with a `Login` class, which has a `+LoginFrame frame` attribute and methods `+Login getInstance()` and `+getLoginFrame`. A `LoginFrame` class is associated with `Login`.
- Facade:** A `Station/Interface` class acts as a facade, containing methods `setContainer()`, `getDisplayPanel()`, `setOperationPanel()`, and `setRMOS()`. It is associated with `LoginFrame`, `Container`, `JPanel`, and `OperationGUIFactory`.
- Factory:** An `OperationGUIFactory` class has a `+createUserInterface()` method, which is associated with the `Station/Interface` and a `createUserInterface: Factory/Method` note. It is associated with several GUI classes: `AddrRCMGUI`, `UpdateRCMGUI`, `EmptyRCMGUI`, `TrackRCMStatus`, and `Activate`. Each of these GUI classes has a `+Jpanel panel` attribute and a `+JPanel getPanel()` method.
- Builder:** A `RecycleItem` interface defines `+CalculatePrice()` and `+getPrice()`. It is implemented by `Glass` and `Paper` classes. A `Builder` class is associated with `RecycleItem`.
- Iterator:** A `RecycleStation` class contains `+Iterator RCMIterator` and `+RCM[]`. It has methods `Iterator getIterator()`, `+getMachines`, and `+setData`. It is associated with a `RecyclingStation` interface (which has `Iterator getIterator()`) and an `RCMIterator` class. The `RCMIterator` class has `+boolean hasNext()` and `+RCM next()` methods. A `Text` class is associated with `RCMIterator`.
- Observer:** An `RCM` class has `+notifyAll()`, `+setChanged()`, and `+addOberser` methods. It is associated with a `RecyclingMachine` interface (which has `+notifyAll()`, `+setChanged()`, and `+addOberser`) and an `Observer` interface (which has `+update(Observer O)`). The `Observer` interface is implemented by a `Subscriber` class, which is associated with a `CustomInterface` class (which has `+update(Observer O)`).

Singleton Design Pattern (Login):

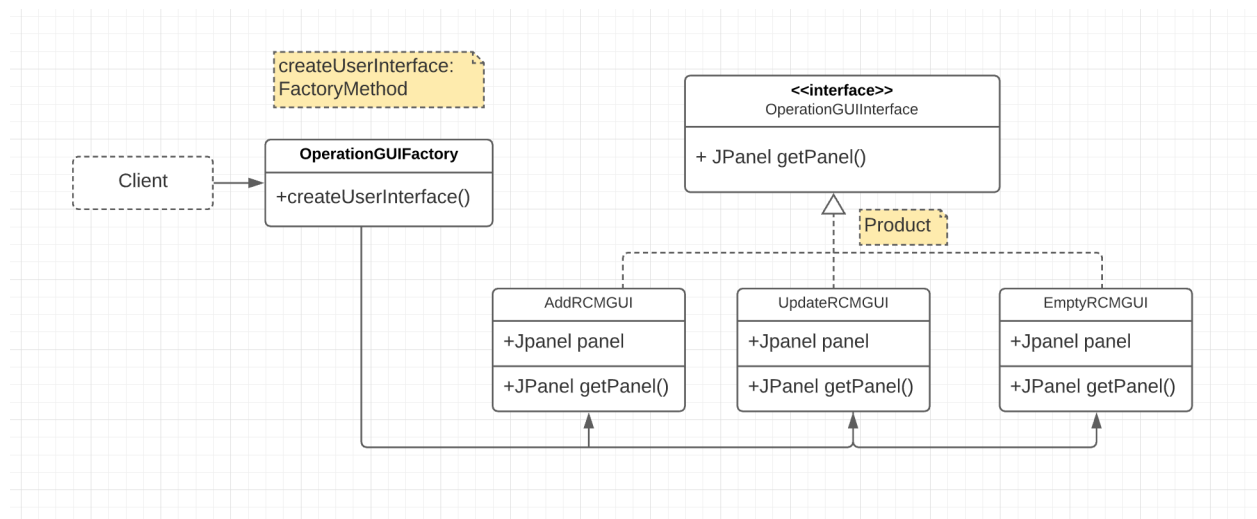
Class Login is a SingletonObject. It will ensure that a class has only one instance and provides a global point of access to that instance. As we will need only one LoginFrame at any moment of time. Please refer UML Diagram for details



Factory Pattern (OperationGUI Factory):

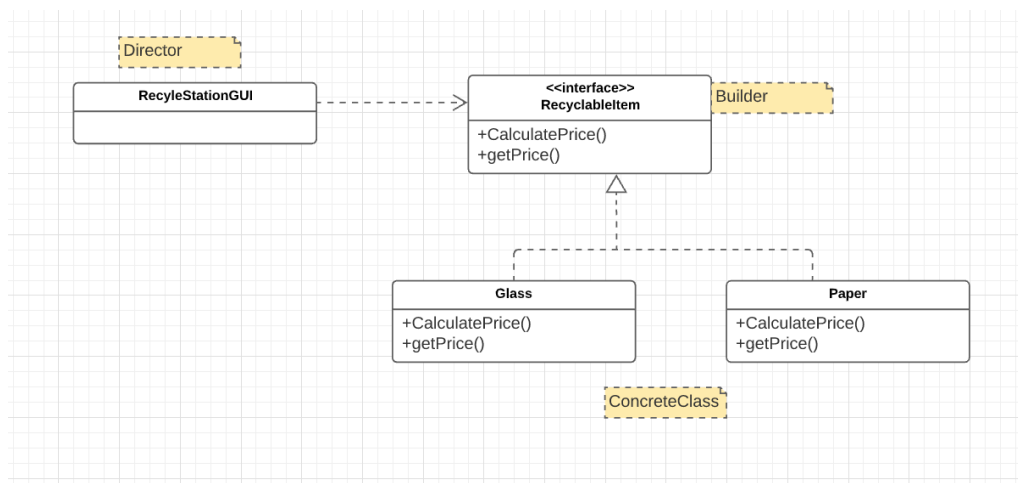
OperationGUIInterface is any interface implemented by many RCM operation user interface screens. I have created a factory in order to accept the operation and return the required operation user interface. Eg for Add/Edit operation the factory will return AddRCMGUI

CreateUserInterface is the factory Method



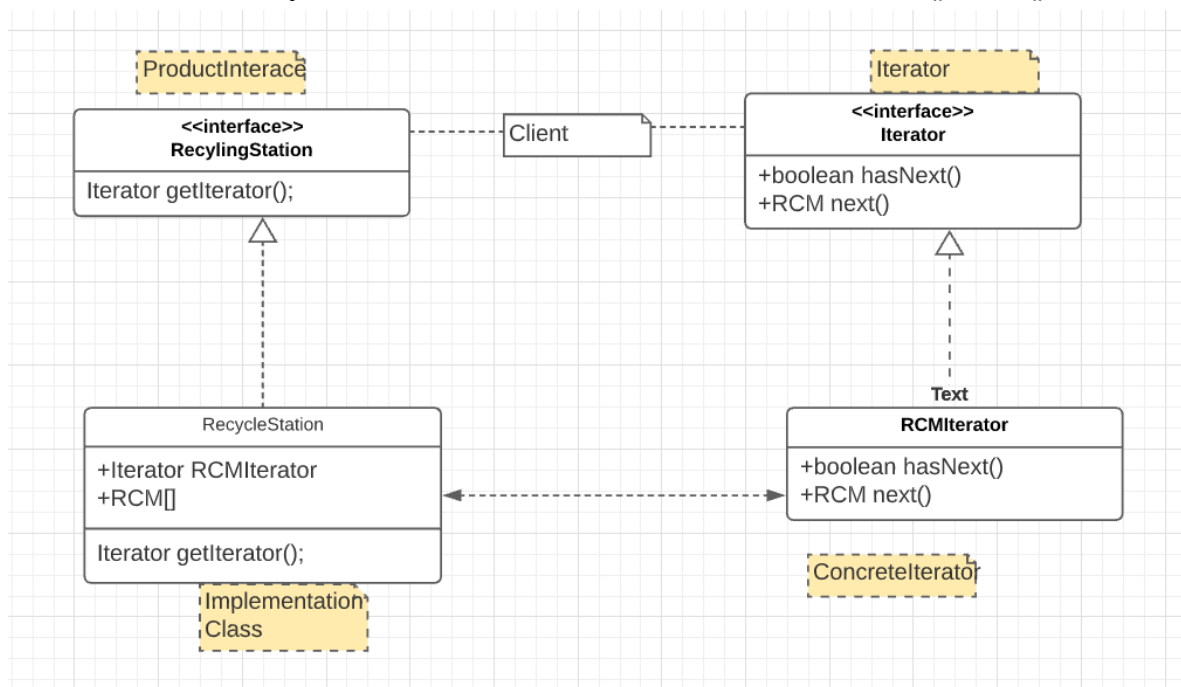
Builder Design Pattern :

I am using a Builder design pattern to construct complex objects of Recyclable items. The pattern will allow me to produce different types and representations of an object using the same construction code. Please refer to the the UML Diagram given below for more details



Iterator Design Pattern:

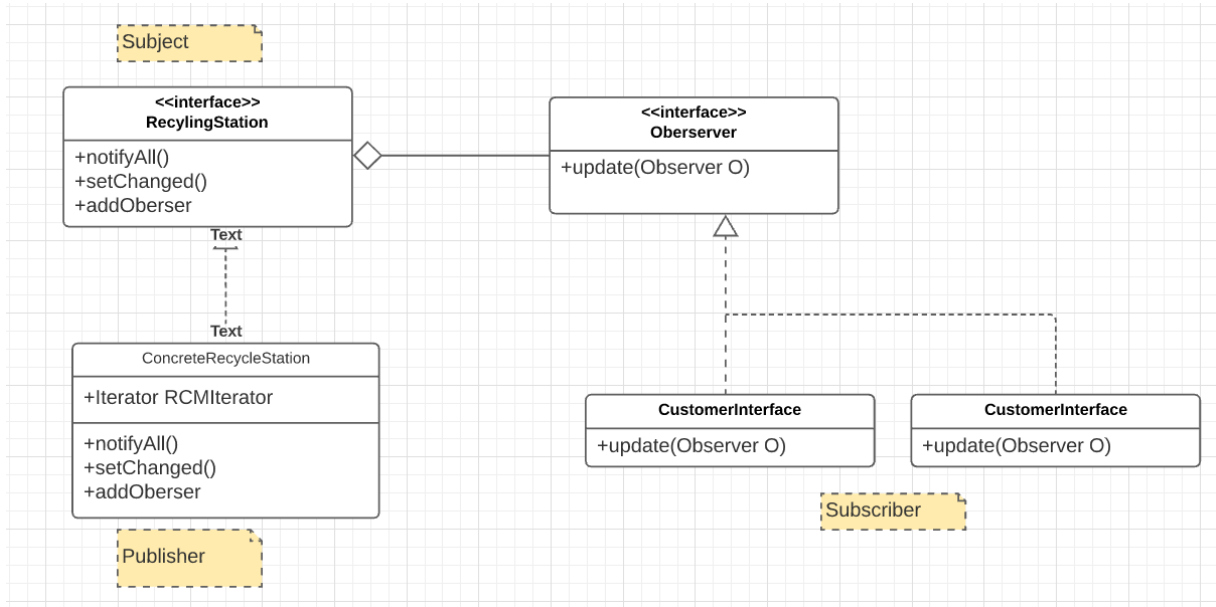
RecyclingStation Class contains an array of RCM. RCMIterator class will help us to iterate over this array. RCMIterator has functions like `hasNext()`, `next()`



Observer Design Pattern:

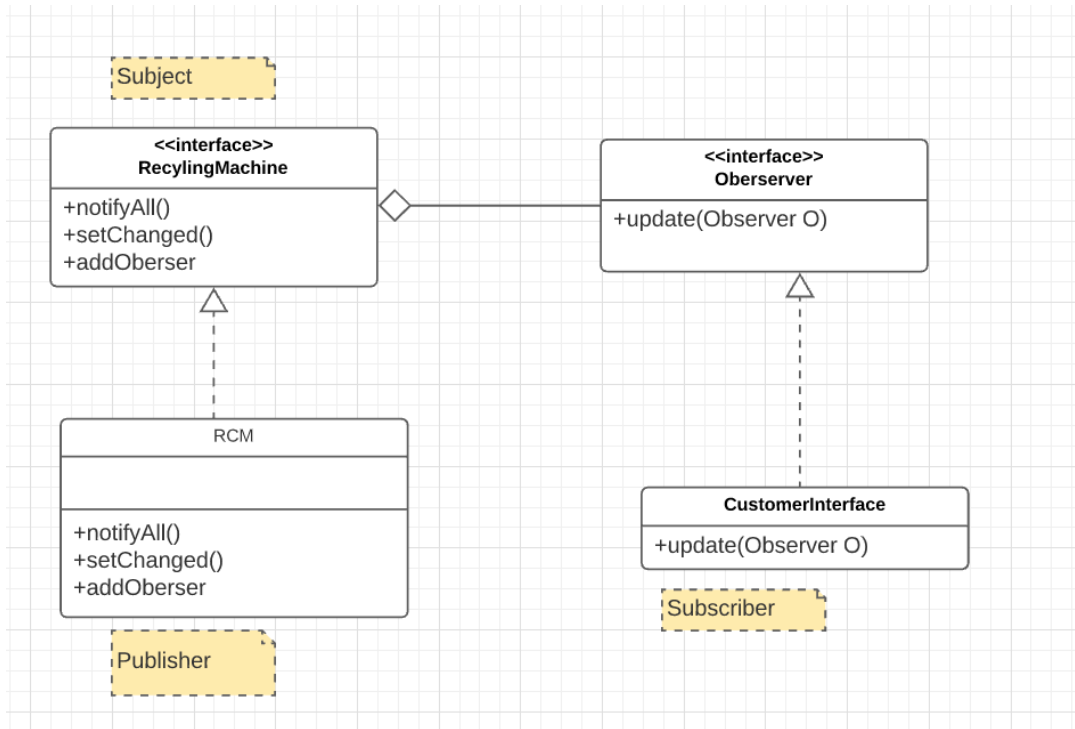
Publisher: Recycling Station

Subscriber: Customer Interface, StationUserInterface



Publisher: Recycling Machine

Subscriber: Customer Interface



Facade Design Pattern(StationUserInterface):

StationUserInterface is a facade class. Here, I am setting up all the required details for the admin view user interface. It has instance of multiple classes and it sets the required data in a particular sequence.

