**Observer Pattern**

Analysis of Pattern

Observer pattern is very useful when there is state management in application. This pattern basically has subscriber ( observer) and Subject ( publisher ). Subject uses an object to update states coming from observer ( members ) and update state as initiated by member object or class. Each observer would not have detail of other member’s logic but state can be shared in entire application by notifying Subject and back to other observer ( member classes). Subject uses observer to notify to child classes about state changes. Observer would update state on object which extend observer.

Benefits

* Makes it easier to manage state in applications
* It’s good to implement this in group of objects when there is one to many dependency

Risks of doing this pattern

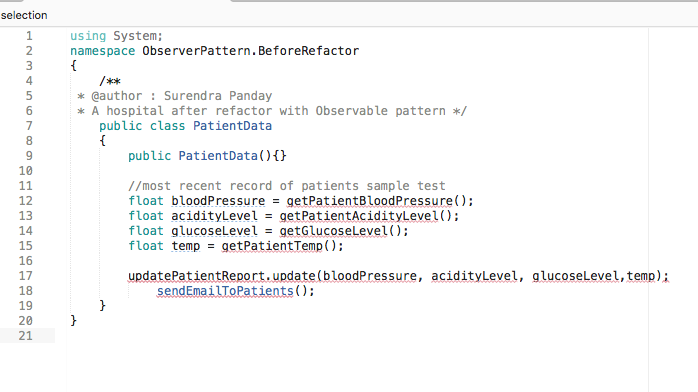
* Observer class is not an interface , it’s basically group of objects talking to each other to notify states so code could have duplicate / code smell, it’s not as clean as using interface

Scenario

A hospital checks patients’ blood pressure, temperature, pulse, height, weight etc. tests and notifies patients about their test results. Patients would not have same data, and they are uniquely identified by their user id, name etc. A hospital would create report for every patient with their statistics from their pressure test, blood test, and so on so forth tests.

With Observable pattern, patients become observer, and hospital becomes subject, and computer system becomes an Object that notifies patients about their test report.

BEFORE REFACTOR



   void sendEmailToPatients();  
            void getPatientBloodPressure() { }  
            void getPatientAcidityLevel() { }  
            void getGlucoseLevel() { }  
            void getPatientTemp() { }

AFTER REFACTOR

A subject that patient data will implement some methods from subject



A class with observer pattern that implements subject interface



**UML Diagram**

check observerPattern.png in ObserverPattern folder

**or**

**https://app.creately.com/diagram/QUaEZAyWDTw/edit**