COMP 345 Fall 18 Week 7

Haotao Lai (Eric)
h_lai@encs.concordia.ca

Lab Instructor

Section: B-X 9999 --W---- 20:30 22:20 H929

Name: Haotao Lai (Eric)

Office: EV 8.241

Email: h_lai@encs.concordia

Website: http://laihaotao.me/ta

Content

- Observer Pattern
- Model-View-Controller

Observer Pattern

Observer Pattern

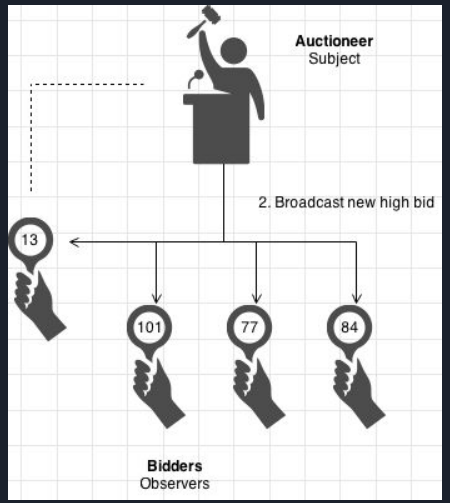
The Observer design pattern is one of the twenty-three well-known "Gang of Four" design patterns that describe how to solve recurring design problems to design flexible and reusable object-oriented software, that is, objects that are easier to implement, change, test, and reuse.

Observer pattern is one of the most common used design pattern in software development.

Observer Pattern

Subject: The object which is observed by an observer. One subject can have any number of observers.

Observer: The object which is watching a subject. Most of the time, one observer will only watch one subject.



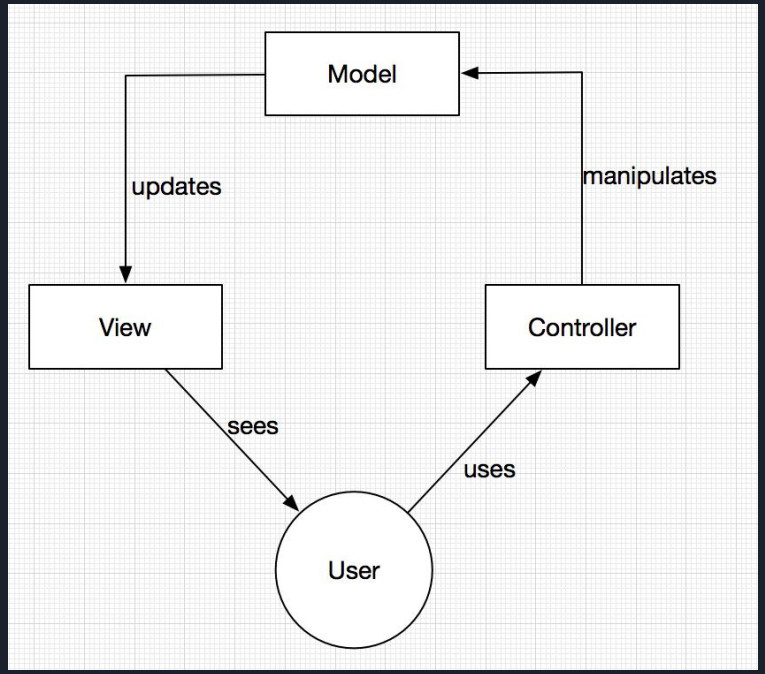
Code Example

```
class Observer {
public:
    virtual ~Observer();
    virtual void update(MessageEnum msg, Subject *pSubject) = 0;
};

class Subject {
public:
    virtual ~Subject();

    virtual void notify(MessageEnum msg) {
        for (auto &it : observerList) {
            it.update(msg, this);
        }
    };

    virtual void attach(const Observer &observer) {
        observerList.push_back(observer);
    };

    virtual void detach(const Observer &observer) {
        observerList.erase(std::remove(observerList.begin(), observerList.end(), observer),
                           observerList.end());
    };

protected:
    std::vector<Observer> observerList;
};
```

MVC Pattern

MVC

Model View Controller (MVC) is a software architecture pattern (not belong to the 23 design patterns). It separates out the application logic into three different parts, promoting modularity and ease of collaboration and reuse. It also makes applications more flexible and welcoming to iterations.



MVC --- Controller

The job of controller is to capture the user action (directly or handle the event pass by the view). It contains the logic how to serve user's request.

MVC --- Model

Model defines the data structure of the application. It doesn't care how the data will be display to the users (which is the view's job). It provides the interface for the controller to manipulate the data.

If the state of the data changed, the model will notify the view to make the corresponding change. In order to achieve that we must somehow need to register the view to the model.

MVC --- View

The only job of the view is to display the data to the users. It only handle the user interaction event, like input something, click a button, or some pre-defined gesture. Those user interaction will be interpreted into some specific instruction and passed to the controller. Then perform some operation on the model.

If the model state changed, then the model will notify the view to render the new UI.

MVC Example