Aflevering 1: The Observer Pattern

I4SWD

27/02-2018

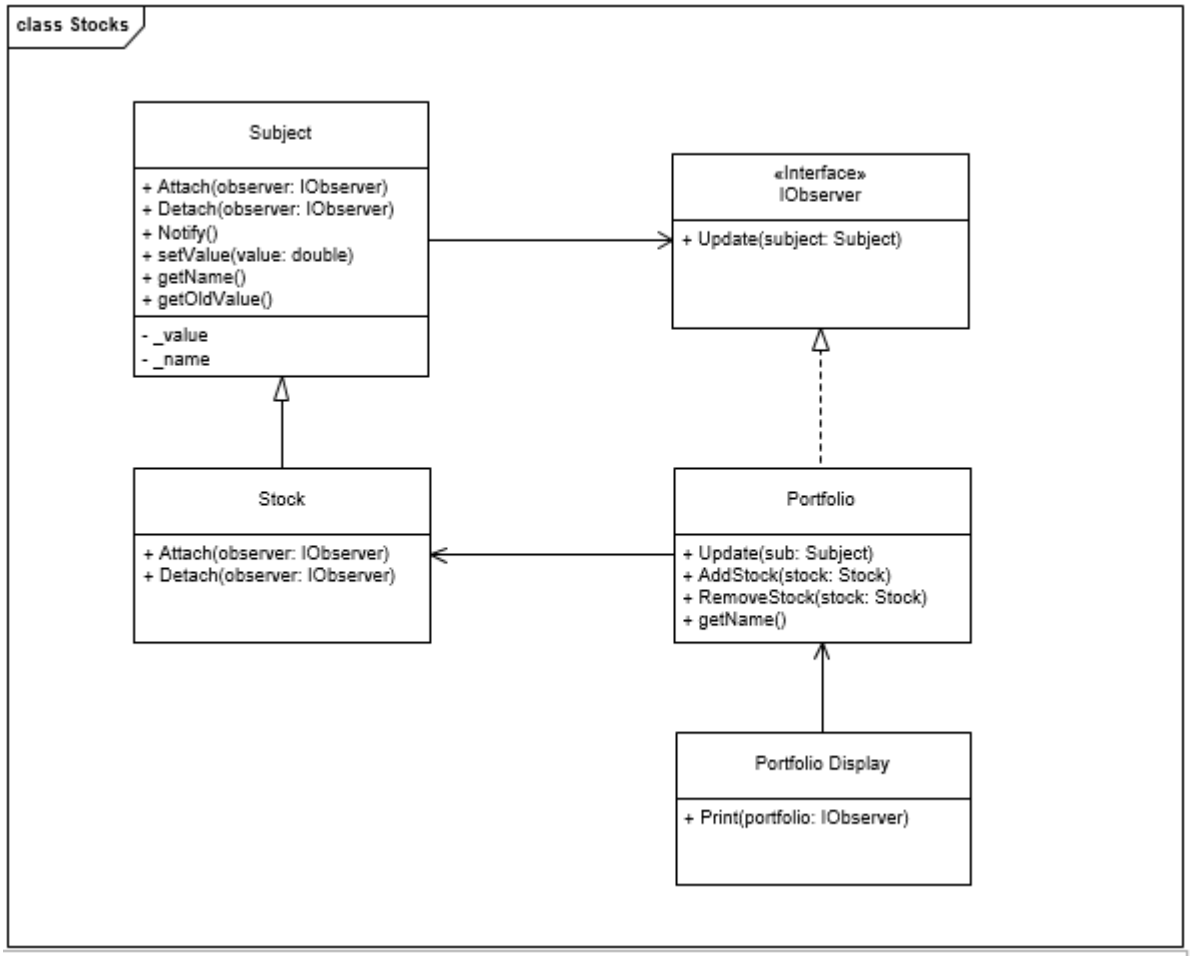
Navn:	Studienr.:
Fatima Kodro	201609565
Martin Haugaard Andersen	201605036
Søren Bech	201604784

Exercise 1: Considering the GoF Observer pattern, what is/are the subject(s), and what is/are the observer(s) in the stock trading system? Which variant of GoF Observer is applicable – or would you rather create your own?

- Stocks = subject
- Publisher = observer
- GoF Observer variant is “push”, because the publisher needs to update automatically.

Exercise 2: Design a system in which Stocks may be added to a Portfolio, which should then automatically be notified if the value of the Stock changes. When this happens, the Portfolio Display should make sure that the stocks in the portfolio are printed to screen.

The class diagram has 5 classes: Subject, Observer, Stock, Portfolio, Portfolio Display.



Subject is an abstract base class for all data subjects, which are the things that need to be updated. In this case it is the Stocks. It has the functions attach(Portfolio) and detach(Portfolio), which will attach/detach the current stock to a specific portfolio. Notify() is called automatically when a change has been made, and it then makes sure to call the Update() function in the IObserver interface which will update the change.

After making the implementation it was discovered, that it was more sufficient to have the Subject class be in charge of the value and name of the stock, instead of making the Stock class itself be in charge of these variables.

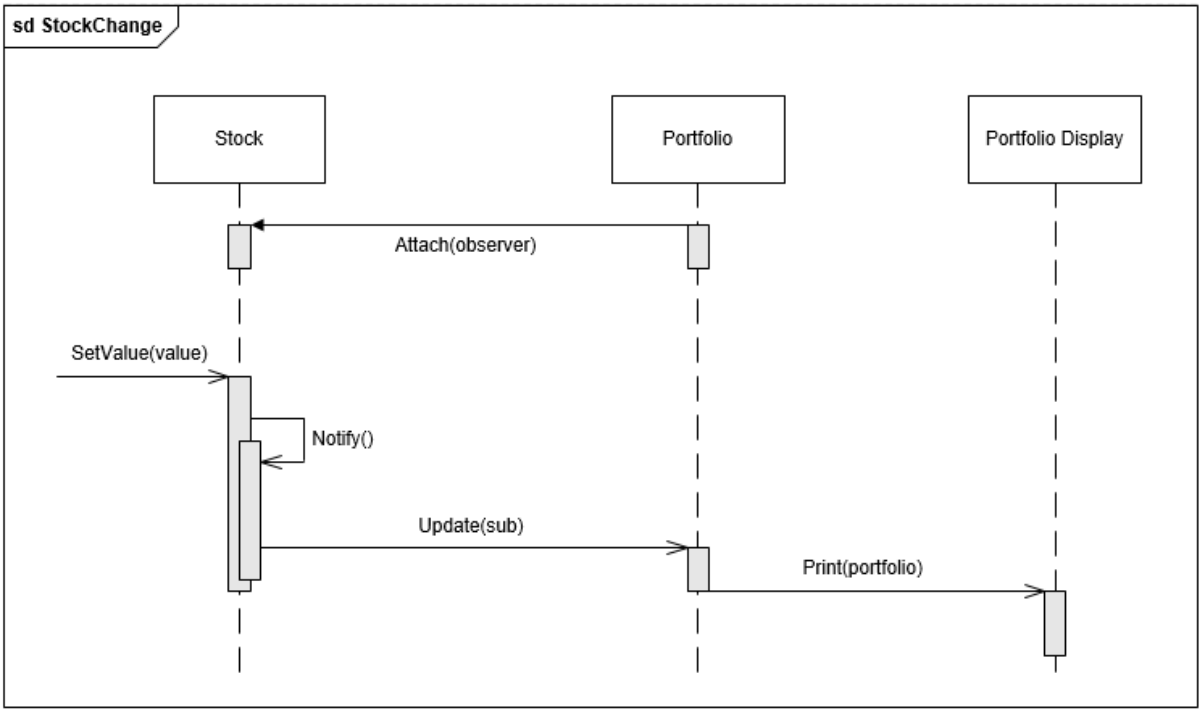
IObserver is an interface that makes sure, that all the classes, that need be informed of data changes, get updated. This happens every time a change happens to the stocks, mainly if the value of the stock is changed.

Stock inherits the attach() and detach() funktions from Subject, which will allow a stock to be attached/detached to a portfolio. This is the actual class that is monitored.
As explained above in the Subject class, after the implementation it was discovered that the Stock class only has to be in charge of attaching and detaching a new stock object, whereas Subject is in charge of when a certain stock changes its value.

Portfolio implements the Observer interface. This is the actual class that must receive updates. To do this, it inherits the Update() function from the Observer class.
After making the implementation it was discovered that this class also needs to have AddStock() and RemoveStock() functions. These functions make sure that every time a stock is attached to a portfolio, the portfolio will also be attached to the stock, making sure they both know about each other.

Portfolio Display outputs the information of the currently held portfolio.

The sequence diagram shows how the sequence of the program is going to be implemented:



First a stock needs to be attached to a specific portfolio. After that, when a new value has been added to the stock by using SetValue(value), the Subject will call the notify(), and since the GoF Observer variant is “push”, the Portfolio will automatically call its Update function. At

last when Portfolio is updated, the Portfolio Display will output information of the currently held portfolio.