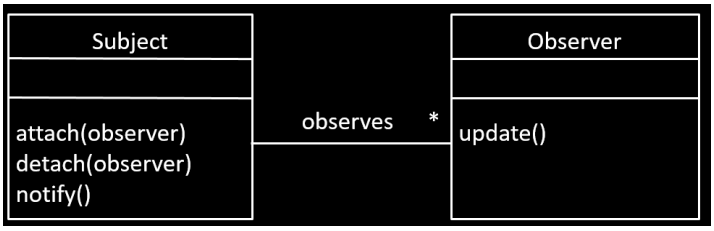
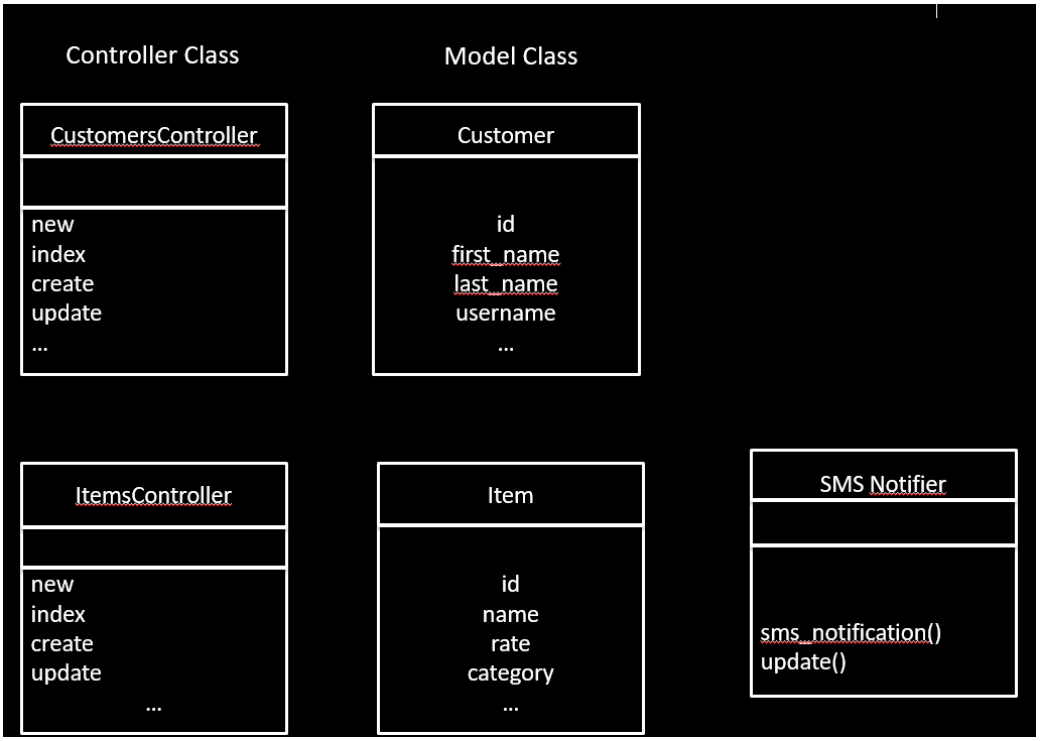
Multiple-Choice Questions:





1. When a customer visits a product on eBay and searches the product multiple times, eBay tracks the search information. Based on this information, eBay sends an SMS notification whenever the product's cost goes down, or there is a discount available for it. Given the Observer design pattern at top, which classes in the diagram below would play the roles of Subject and Observer. (Hint: the system notifies the customer if there is a change in the cost to the item).

    a. Subject: CustomersController and Observer: Item

    b. Subject: ItemsController and Observer: SMSNotifier

    c. Subject: ItemsController and Observer: Item

    d. Subject: CustomersController and Observer: SMSNotifier

    e. None of the above

Solutions:

1. b

2. a

Consider these figures when answering the following question.

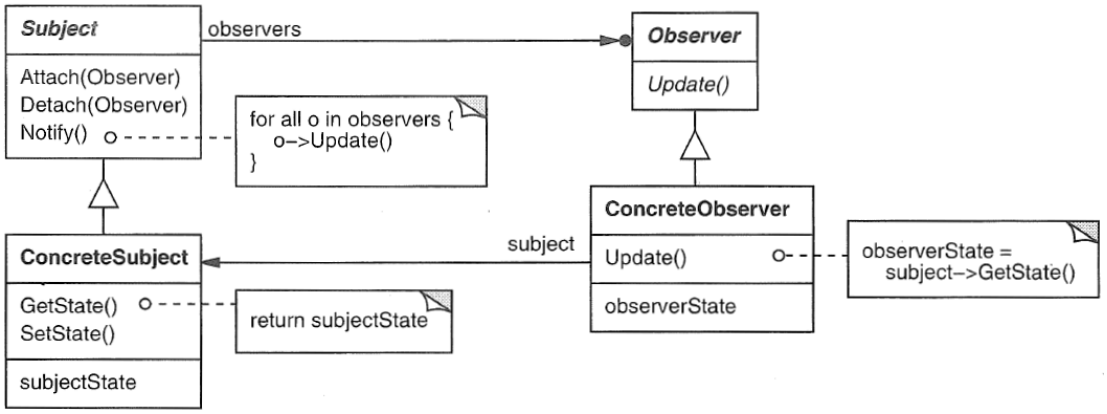


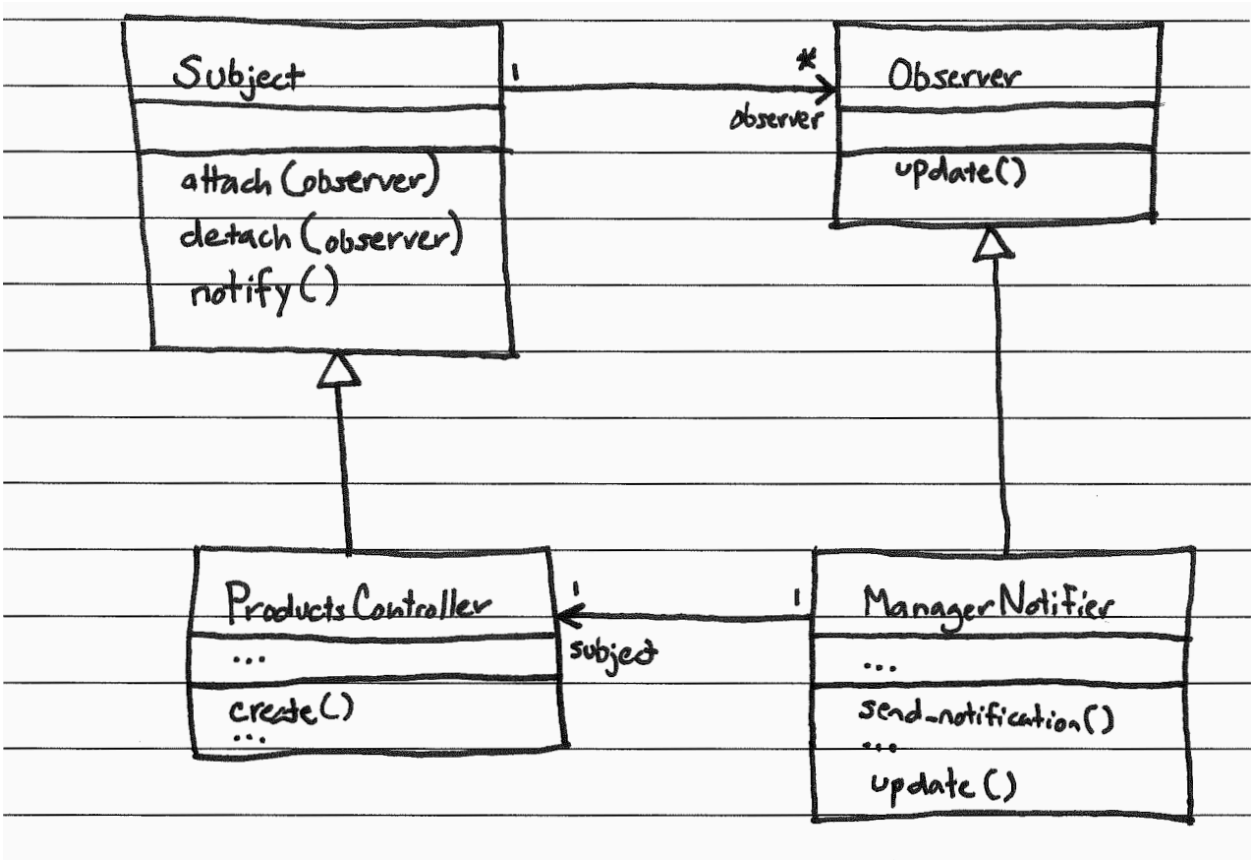
Figure 3. Observer Pattern from the "Gang of Four" book. (Note that the book uses an outdated class diagram notation.)

Figure 4. Classes for product-supply system.

Problem:

Recall the Observer Design Pattern depicted in Figure 3. Imagine that you are designing a web app for product-supply business. Figure 4 depicts the classes that you have so far. In particular, you have designed a ProductsController class that records product information. As part of this controller's responsibilities, it must create new product entries when new products are added. You have also designed a ManagerNotifier that is capable of sending notification messages to managers at the company. The design problem you need to solve is how to make a ManagerNotifier "listen" for when a ProductsController creates a new product, and to send a notification to a manager whenever that happens. Draw a class diagram that applies the Observer Design Pattern to solve this problem. Use the same names used in the design pattern as much as possible (except make Ruby style). You must include all the classes from Figure 4 in your diagram (i.e., your changes should be additive). In particular, I expect that you will be adding classes, operations, inheritance relationships, and associations.

Solution:


Subject
attach(observer)
detach(observer)
notify()
1
*
observer
Observer
update()
ProductsController
...
create()
...
1
1
subject
ManagerNotifier
...
send_notification()
...
update()

Consider these figures when answering the following question.

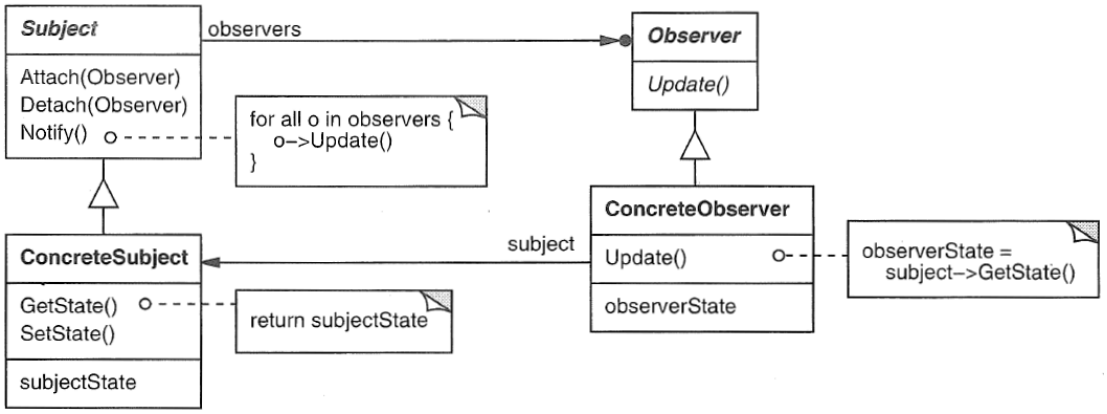


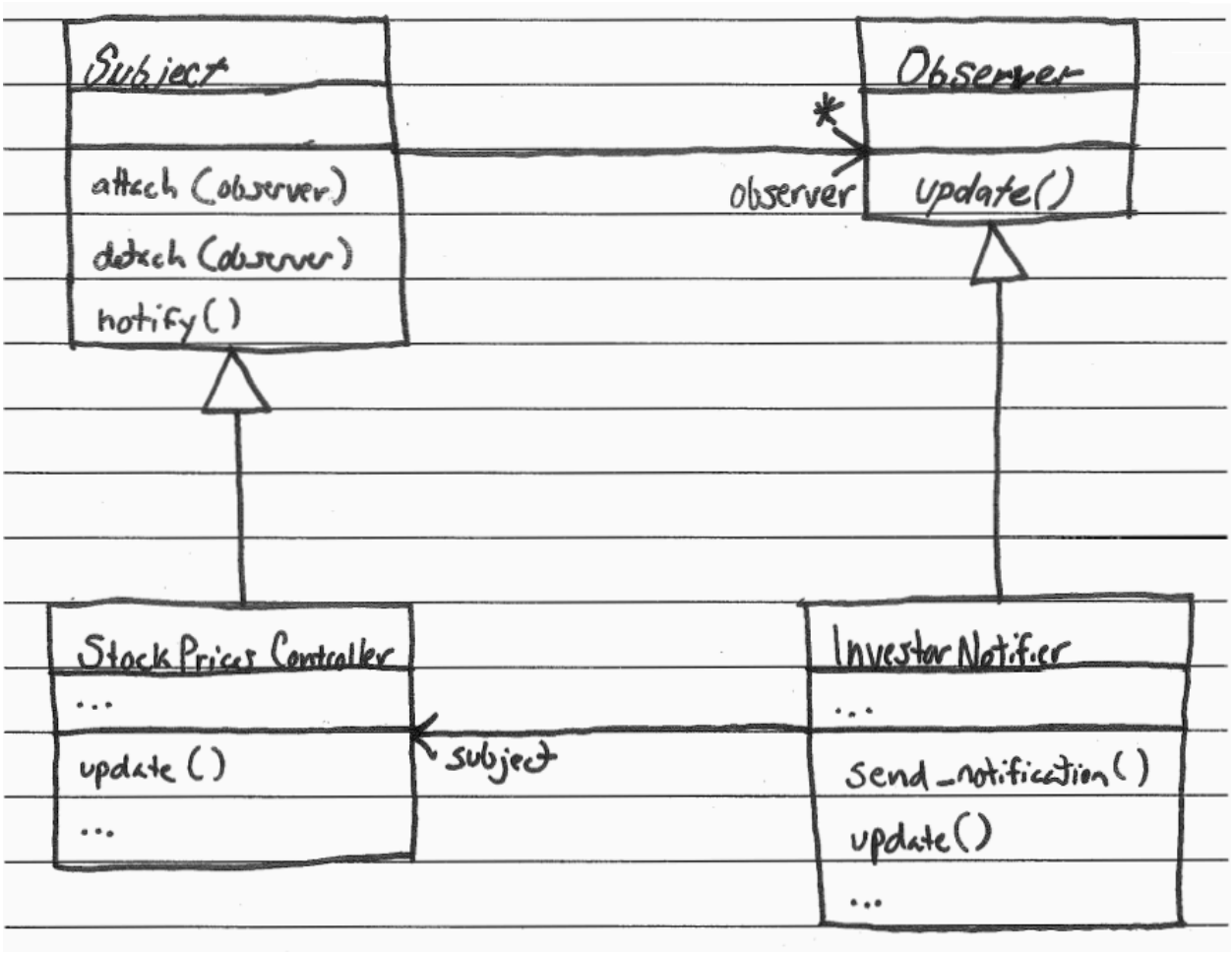
Figure 5. Observer Pattern from the "Gang of Four" book. (Note that the book uses an outdated class diagram notation.)

Figure 6. Classes for investment company web app.

Problem:

Recall the Observer Design Pattern depicted in Figure 5. Imagine that you are designing a web app for an investment company. Figure 6 depicts the classes that you have so far. In particular, you have designed a StockPricesController class that records price changes to stocks. As part of this controller's responsibilities, it must update stock price entries as they change. You have also designed a InvestorNotifier that is capable of sending notification messages to investors. The design problem you need to solve is how to make a InvestorNotifier "listen" for when a StockPricesController updates a stock price, and to send a notification to affected investors whenever that happens. Draw a class diagram that applies the Observer Design Pattern to solve this problem. Use the same names used in the design pattern as much as possible (except make Ruby style). You must include all the classes from Figure 6 in your diagram (i.e., your changes should be additive). In particular, I expect that you will be adding classes, operations, inheritance relationships, and associations.

Solution:


Subject
attach (observer)
detach (observer)
notify()
Observer
*
observer
update()
StockPricesController
...
update()
...
subject
InvestorNotifier
...
send_notification()
update()
...