Observer ID

Student:

Izabela Kuźniar

Teacher:

Andrea Corradini

Course:

Software Design Patterns

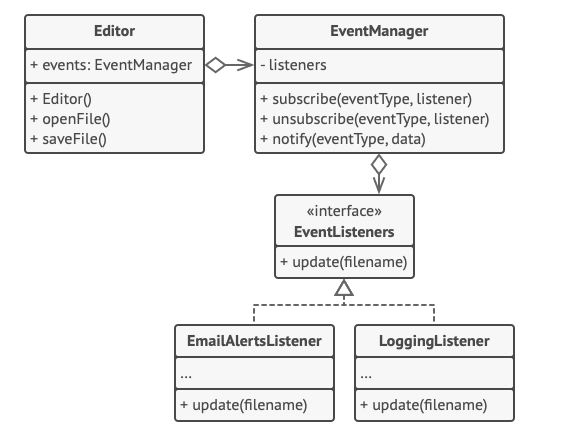
# Name and category

Observer is a behavioral pattern. Also known as Publish-Subscribe pattern.

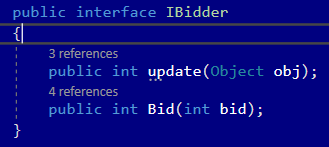
# Intent:

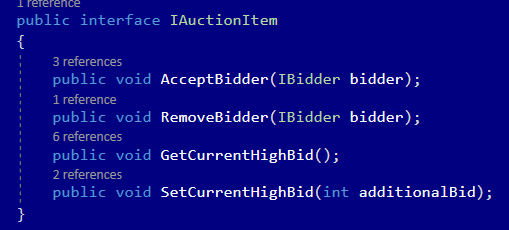
Able to define subscription to notify all the subscribers of certain object that they are observing.

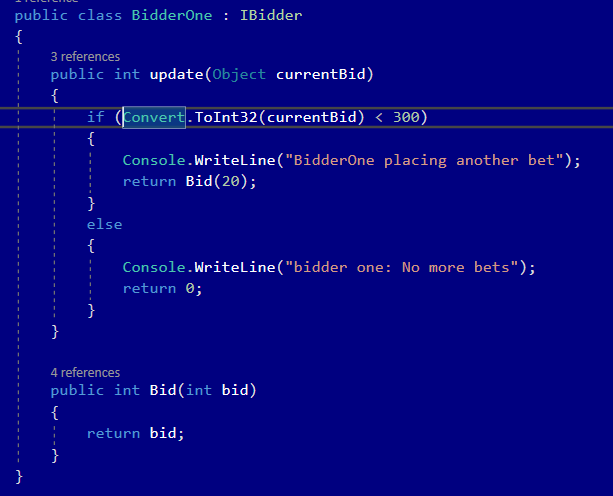
# Structure as a UML class diagram

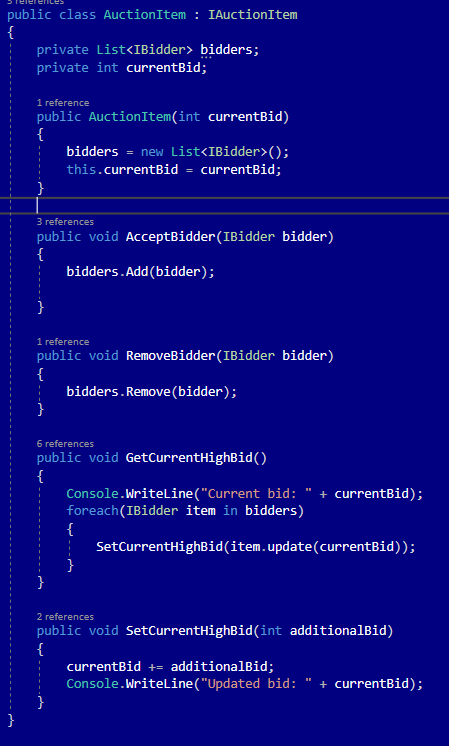


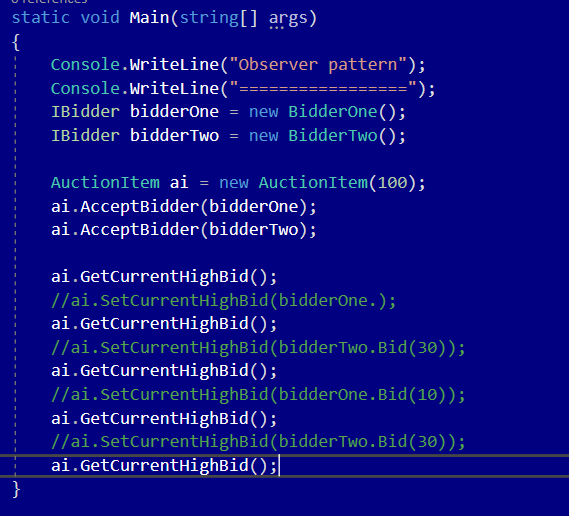
# Implementation:











# Consequences:

Benefits:

* Open/Closed Principle. Ability to introduce new subscriber classes without having to change the publisher’s code (and vice versa if there’s a publisher interface).
* Ability to establish relations between objects at runtime.

Drawbacks:

* Subscribers are notified in random order.

# Known uses

* Swing/GUI and listeners
* Java Message Service
* MVC framework

# Related patterns

1. Chain of Responsibility, Command, Mediator and Observer address various ways of connecting senders and receivers of requests:
   1. Chain of Responsibility passes a request sequentially along a dynamic chain of potential receivers until one of them handles it.
   2. Command establishes unidirectional connections between senders and receivers.
   3. Mediator eliminates direct connections between senders and receivers, forcing them to communicate indirectly via a mediator object.
   4. Observer lets receivers dynamically subscribe to and unsubscribe from receiving requests.
2. The difference between Mediator and Observer is often elusive. In most cases, you can implement either of these patterns; but sometimes you can apply both simultaneously.