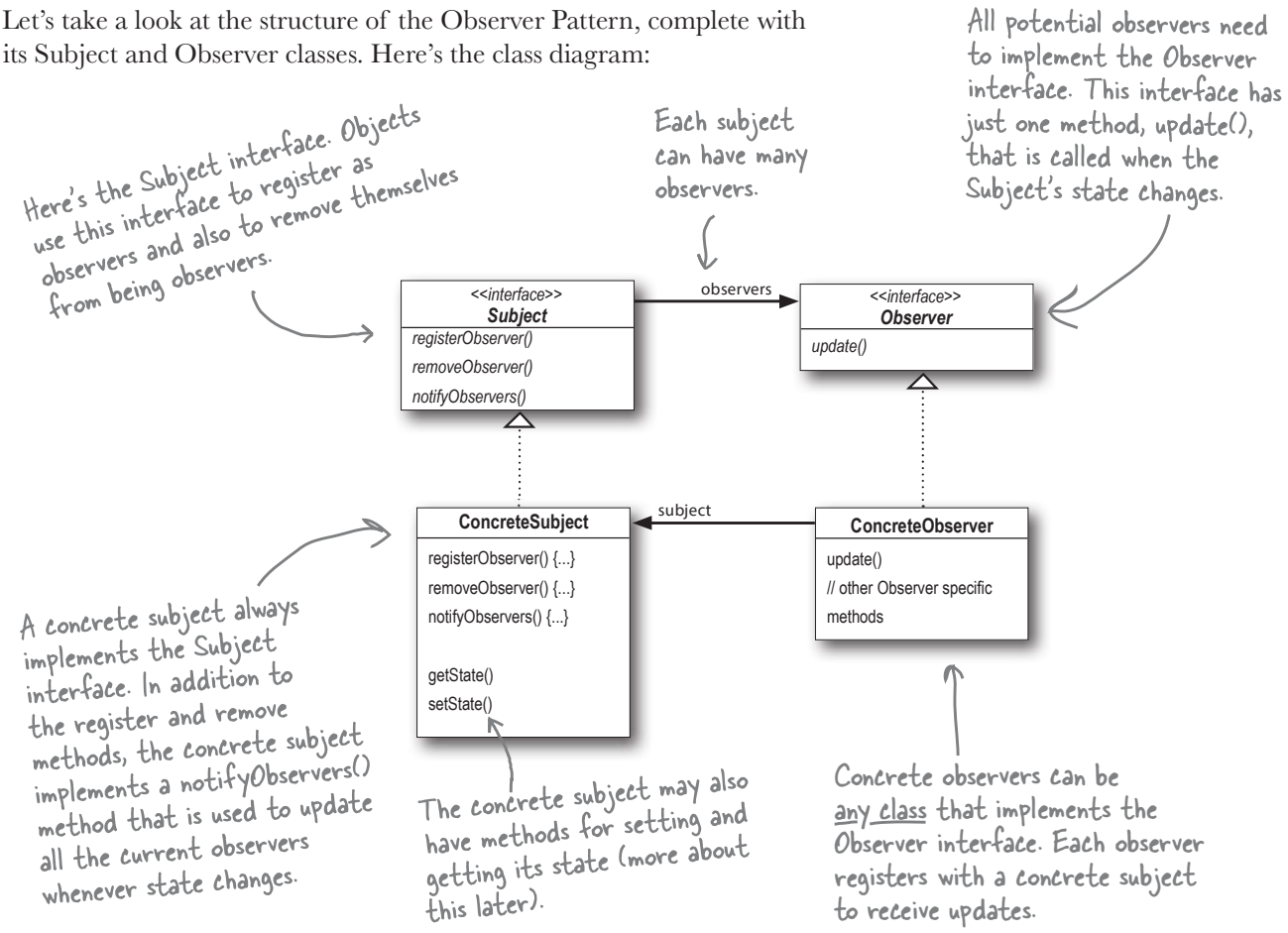
The Observer Pattern: the Class Diagram

Let's take a look at the structure of the Observer Pattern, complete with its Subject and Observer classes. Here's the class diagram:



there are no Dumb Questions

Q: What does this have to do with one-to-many relationships?

A: With the Observer Pattern, the Subject is the object that contains the state and controls it. So, there is ONE subject with state. The observers, on the other hand, use the state, even if they don't own it. There are many observers, and they rely on the Subject to tell them when its state changes. So there is a relationship between the ONE Subject to the MANY Observers.

Q: How does dependence come into this?

A: Because the subject is the sole owner of that data, the observers are dependent on the subject to update them when the data changes. This leads to a cleaner OO design than allowing many objects to control the same data.

Q: I've also heard of a Publish-Subscribe Pattern. Is that just another name for the Observer Pattern?

A: No, although they are related. The Publish-Subscribe pattern is a more complex pattern that allows subscribers to express interest in different types of messages and further separates publishers from subscribers. It is often used in middleware systems.