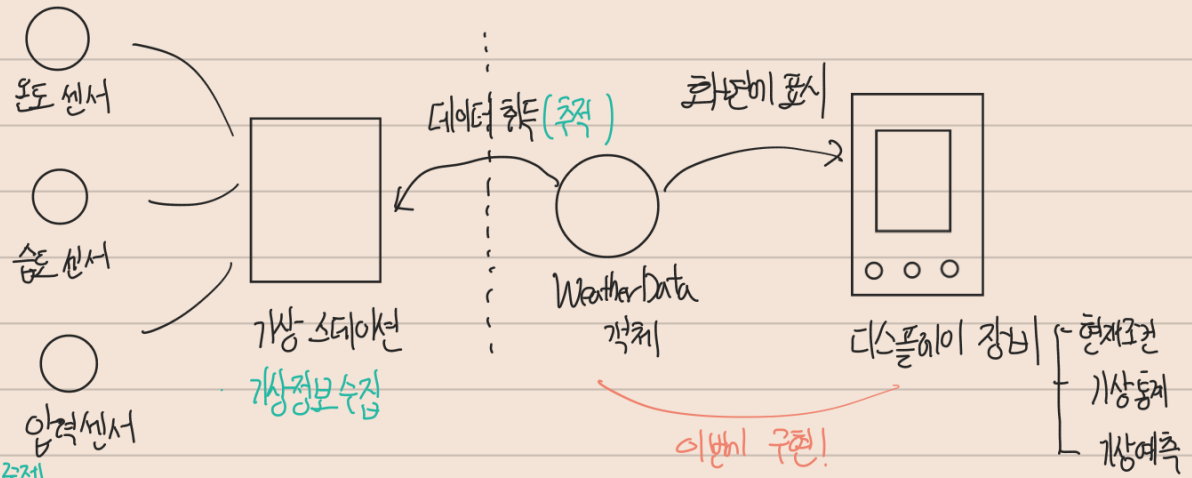
기상 모니터링 앱



주제 Subject 옵저버 observer

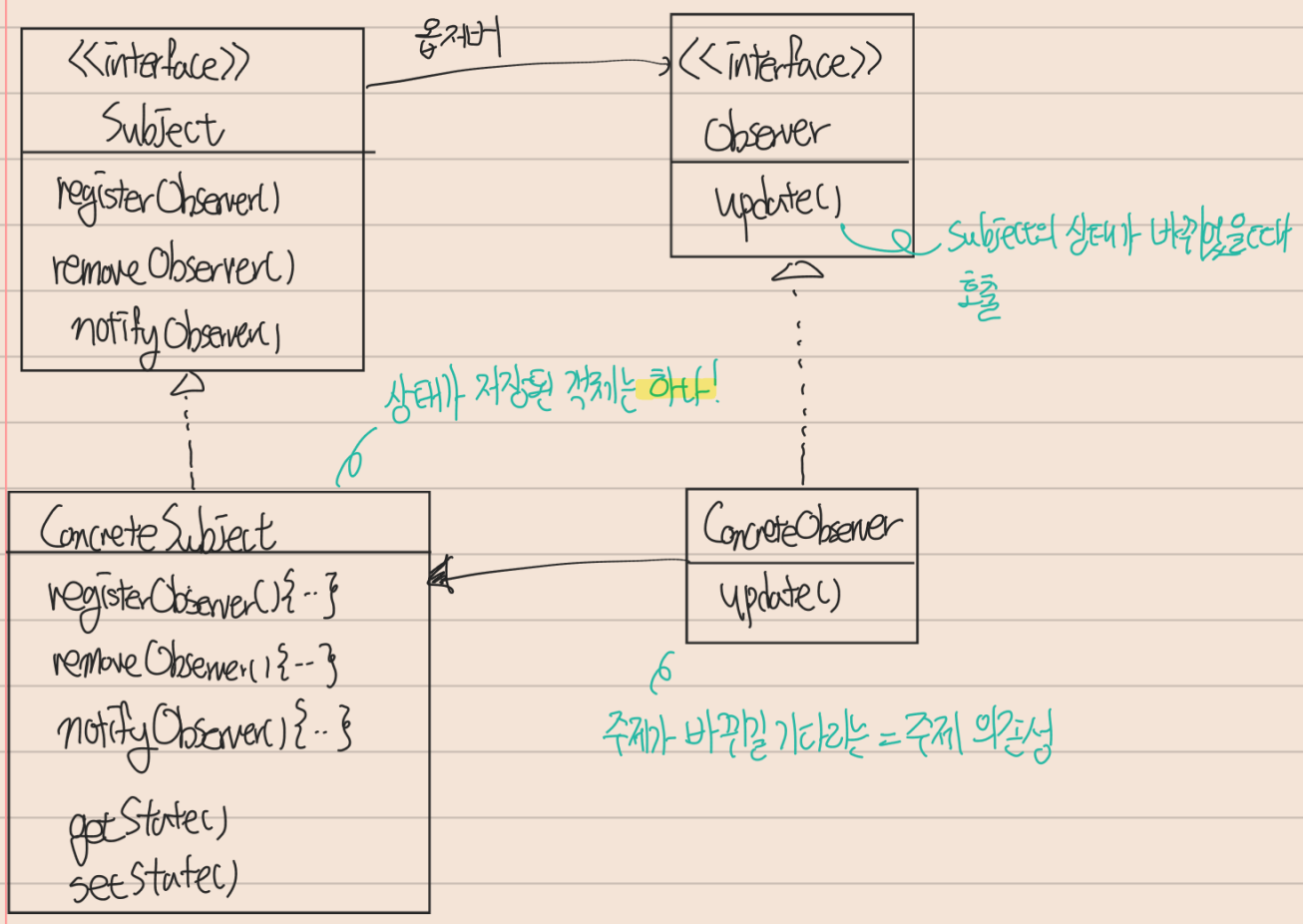
- 출판자 + 구독자 = 옵저버 패턴

옵저버 패턴

옵저버는 주제에 의존함

: 한 객체 상태가 바뀌면 그 객체에 의존하는 다른 객체들한테 연락이 가고 자동으로 내용이 갱신되는 방식으로 일대다(one-to-many) 의존성을 정의합니다.

- 대부분 주제 인터페이스와 옵저버 인터페이스가 들어있는 클래스 디자인을 바탕으로 함



⇒ 여러 객체에서 동일한 데이터를 제어하기보다 훨씬 깔끔!

- 주제와 옵저버는 서로 독립적으로 재사용 가능
- 주제가 옵저버에 대해 아는건 Observer Interface를 구현하는 것 뿐

디자인 원칙

서로 상호작용하는 객체 사이에서는 가능하면 느슨하게 결합하는 디자인을 사용해야한다.

Observer class 생성자 / 생성자 주입

```
public CurrentConditionsDisplay(Subject weatherData) {
    this.weatherData = weatherData;
    weatherData.registerObserver(this);
}
```

↳ 레퍼런스를 저장해두면 옵저버 탈퇴시 유용하게 쓸 수 있음.

Q. 생성자 주입하면 Subject 없이는 생성 못함. 느슨한 결합이 아닌것 같은데?

Observable

- Subject ⟶ Observer (push 방식)
- Subject ⟵ Observer (pull 방식) ↳ 더 옳은 것으로 간주됨. Subject가 변화했다는 noti를 날리면 Observer에서 noti를 받고 필요한 정보만 Subject의 getter를 통해서 읽어오기.

- java.util 패키지에 Observer interface와 Observable 클래스도 있음 (102p)

⚠ 옵저버에게 연락하는 순서에 의존하면 절대 안됨! 느슨한 결합이 아님!