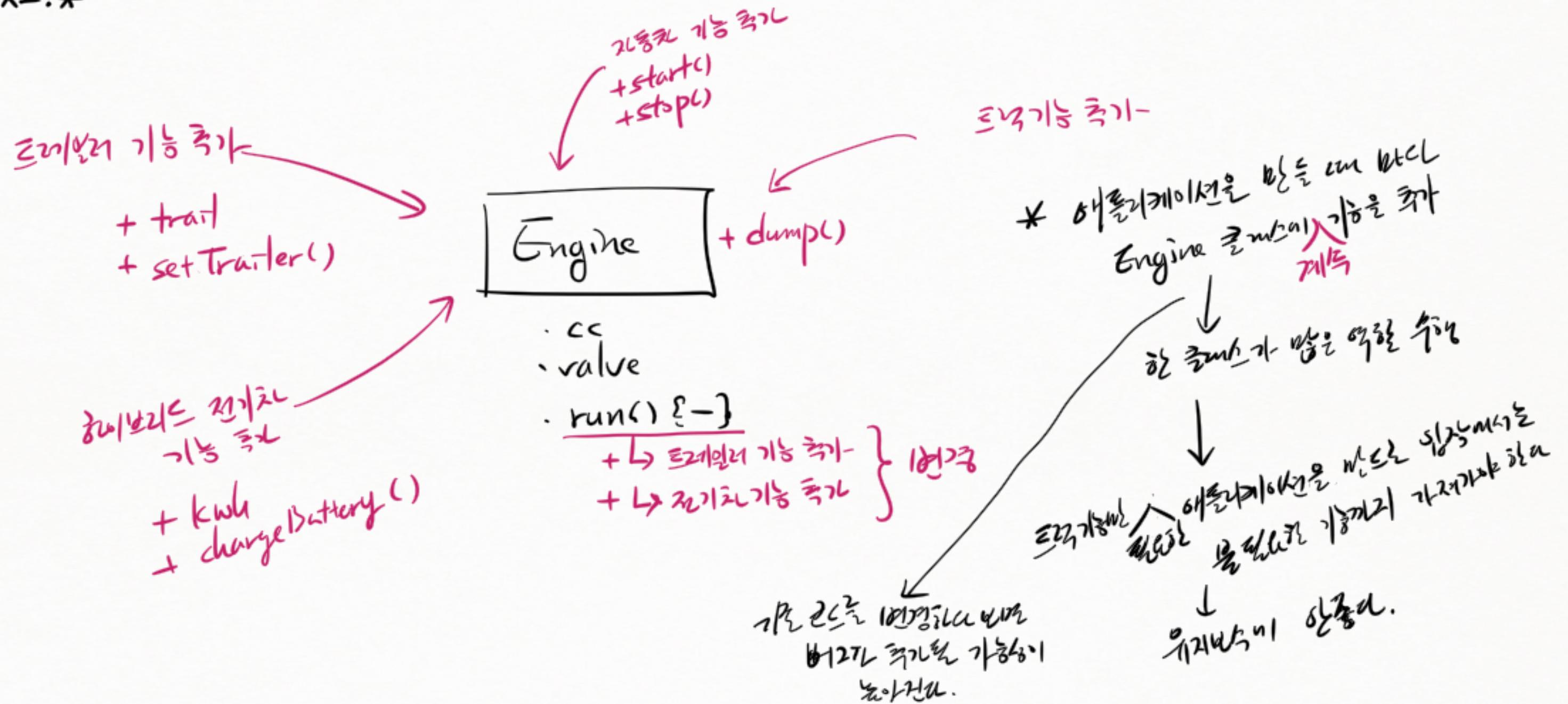


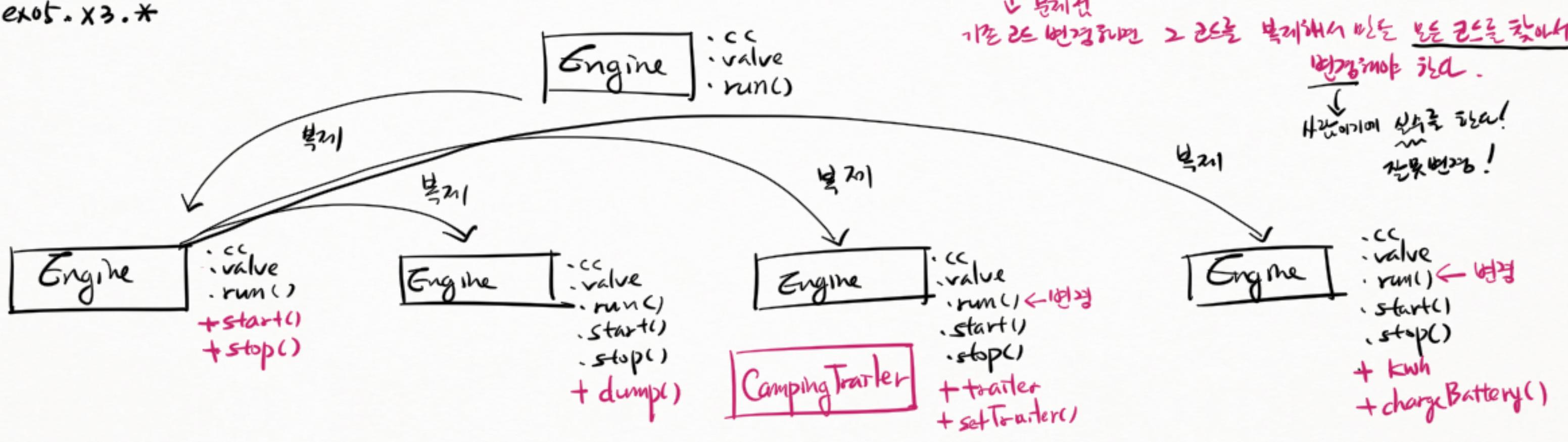
* 기능 확장 방법 1 - 기존 코드에 새 기능 추가

oop.ex5.x2.*



* 기능 확장 방법 2 - 복제한 코드에 새 기능 추가

oop.ex05.*



① 자동차 만들기
app1

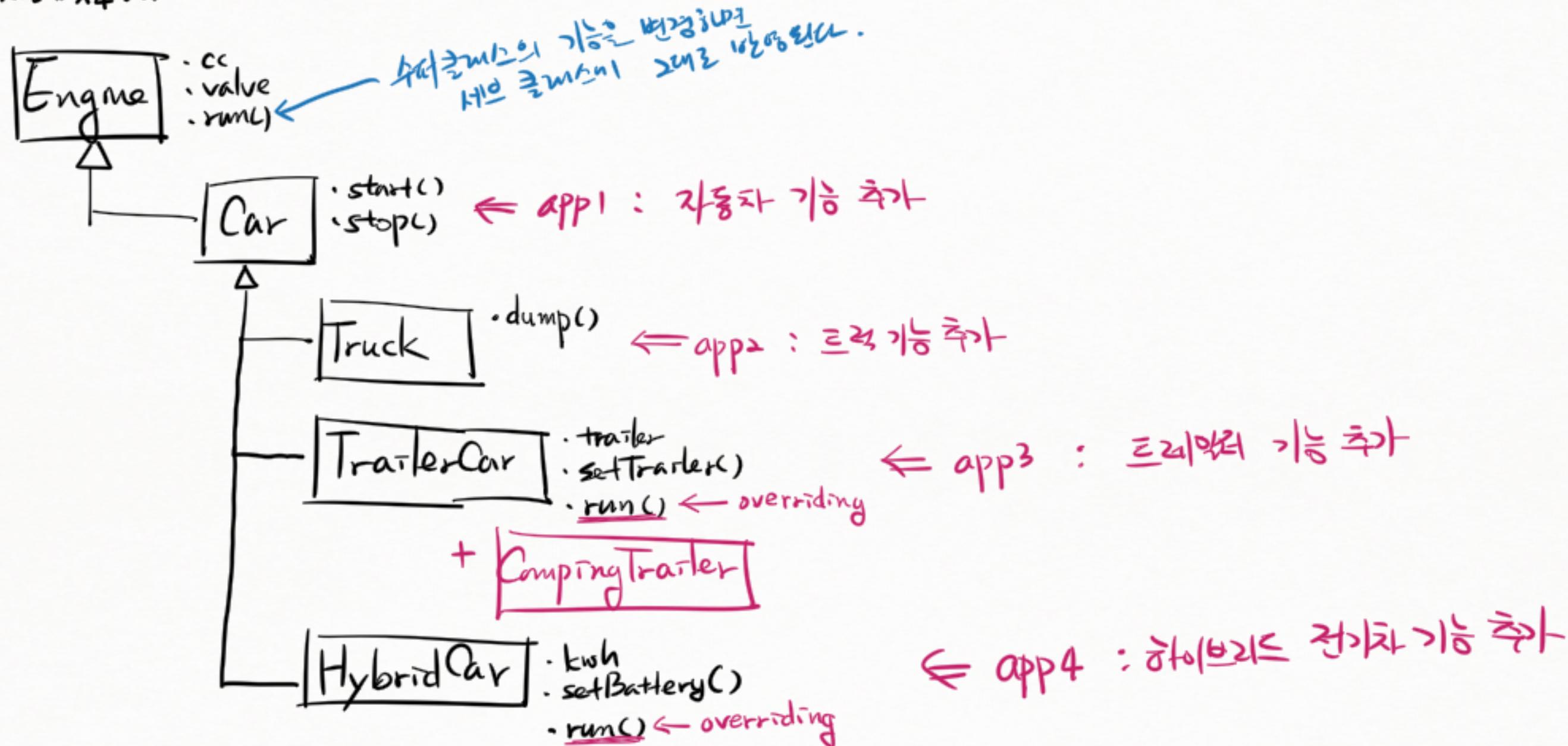
② 트레일러 만들기
app2

③ 캠핑카 만들기
app3

④ 차량으로 전기차 만들기
app4

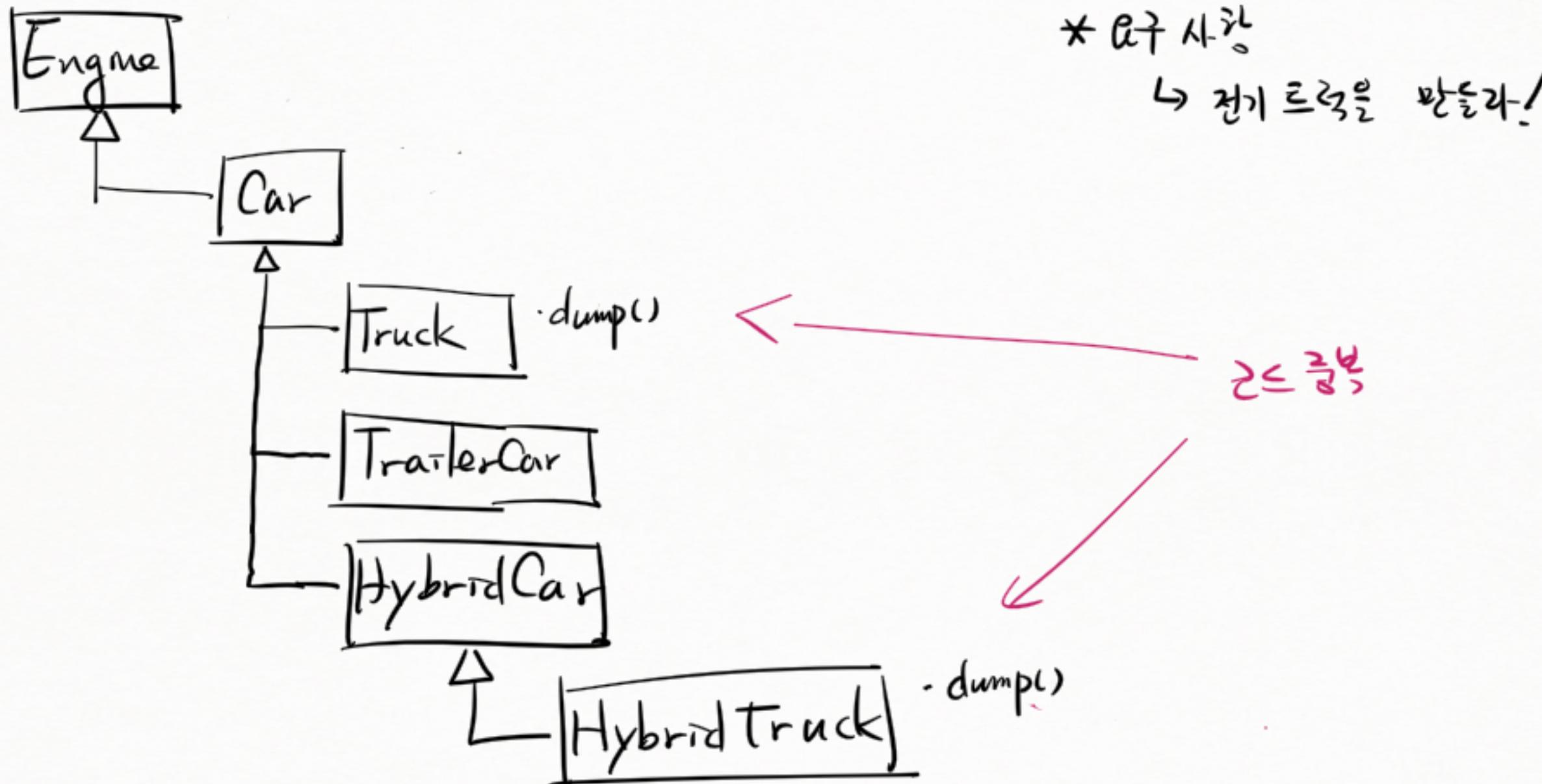
* 기능 확장 방법 3 - 상속을 이용한 확장.

oop.ex05.x4.*



* 기능 확장 방법 3 - 상속을 통한 기능 확장의 한계

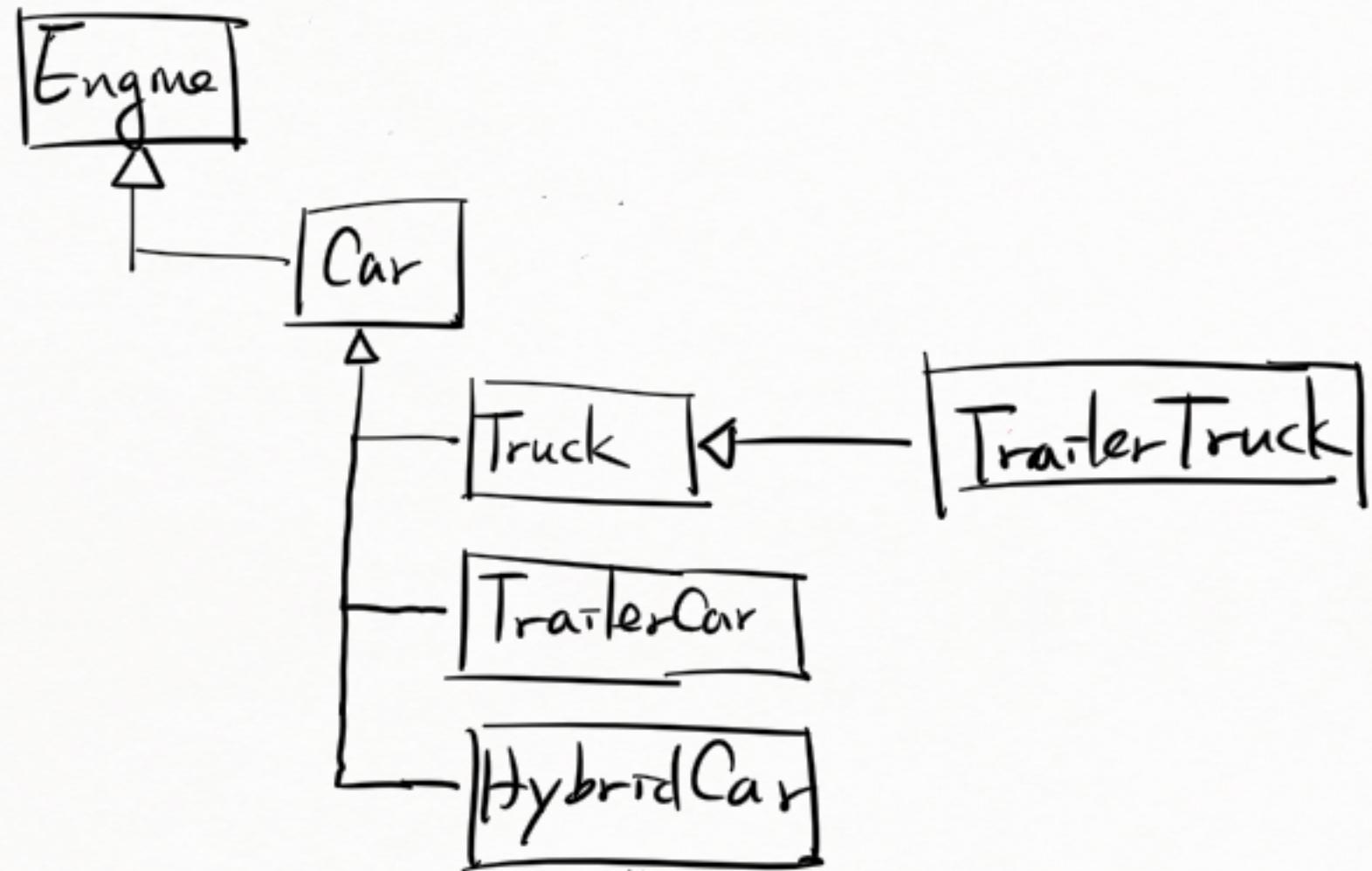
oop.ex05.x4.*



* 기능 확장
↳ 전기 트럭을 만들라!

* 기능 확장 방법 3 - 상속을 통한 기능 확장의 한가지

oop. ex05. x4.*



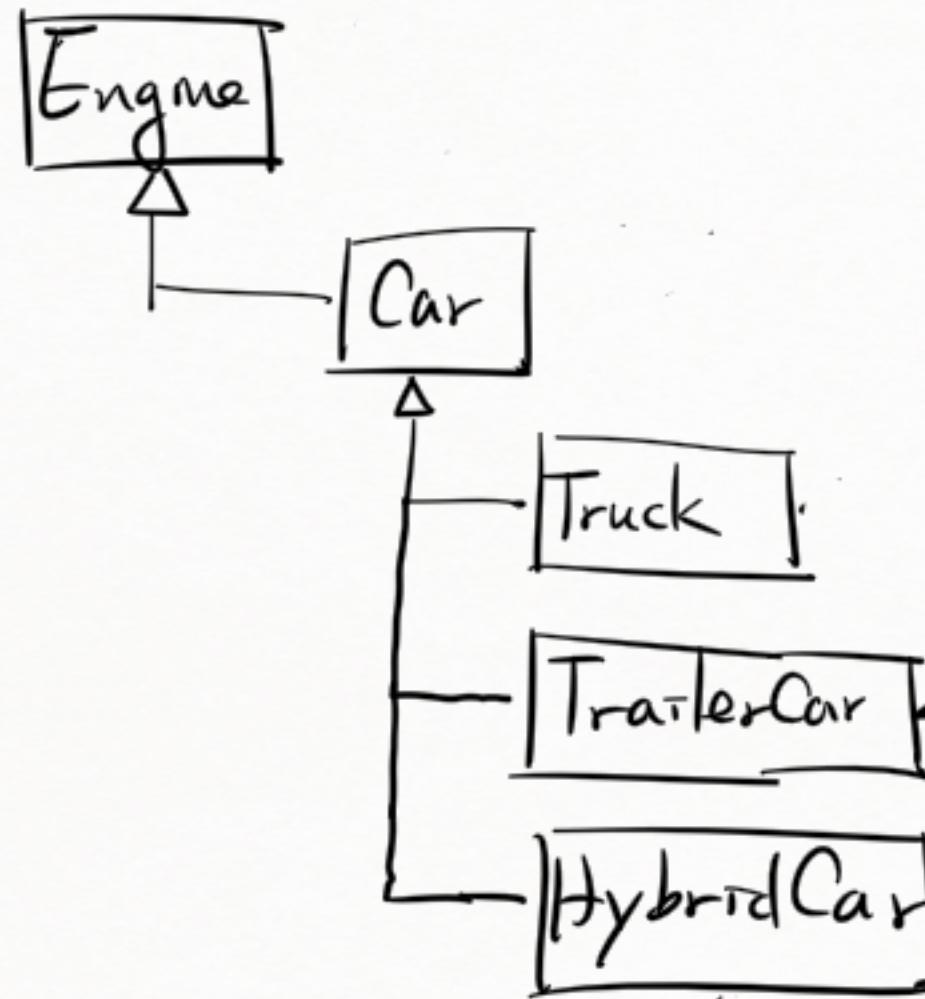
* 추상화

↳ 트레일러를 뒤에 두 수 있는 트럭

· trailer
· setTrailer()

* 기능 학습 18주 3 - 상속을 통한 기능 학습의 한계

oop. ex05. x4.*



* 예시 사용

↳ 전기 캐리어카

* 이렇게 기능 조합을 하려면
수행은 퍼블릭 메소드가 대체된다

- kwh
setBattery()

(상속으로 대상은 기능이 조합된
개체로 만든다면 가능)

↳ 유지보수는 어렵지만

* 기능학적 특성 4 : 디자인에서 기초 기능

기능적

Sedan

Truck

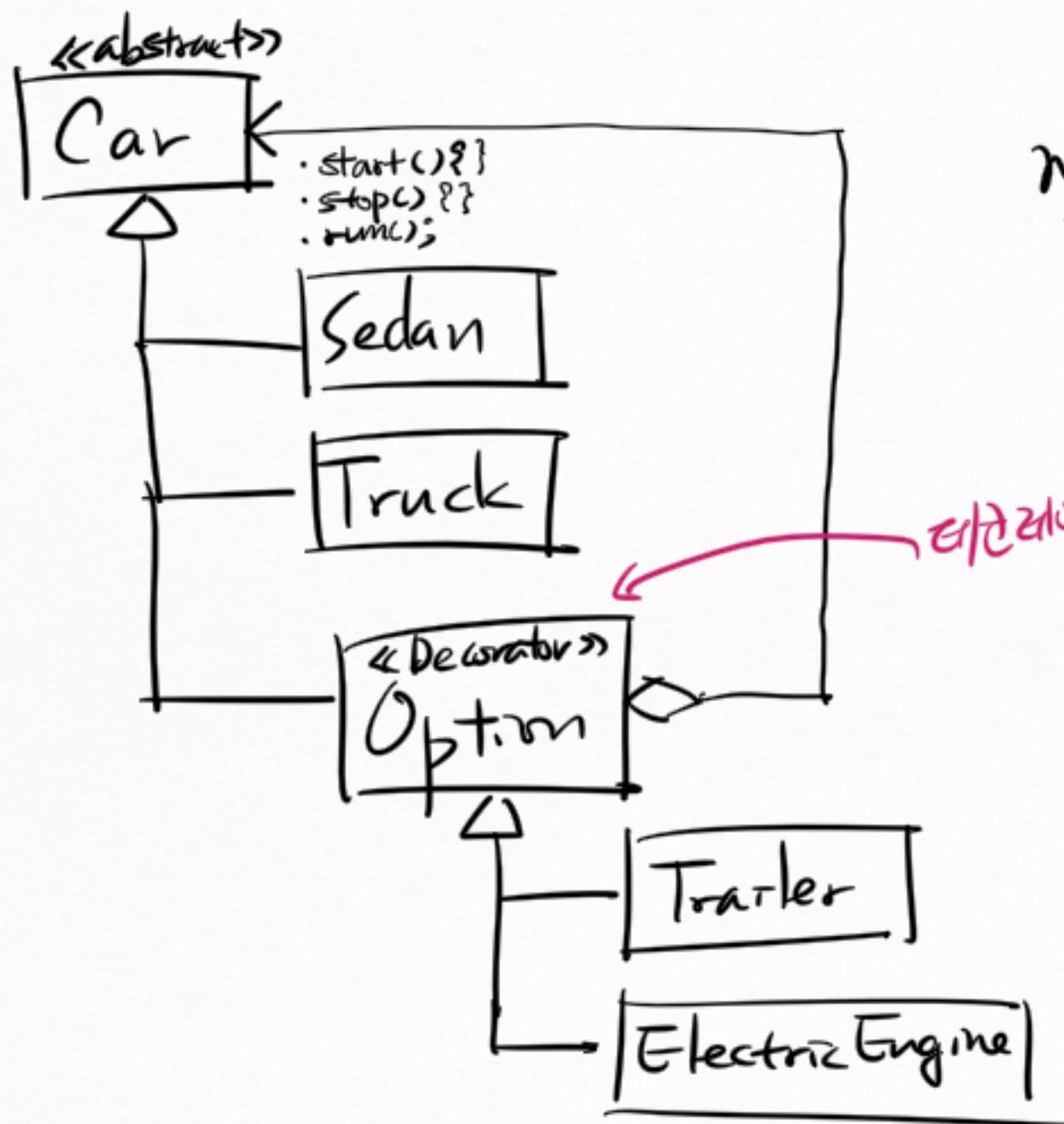
선택적

Gas Engine

Electric Engine

Trailer

* 테러리아 키미 기획

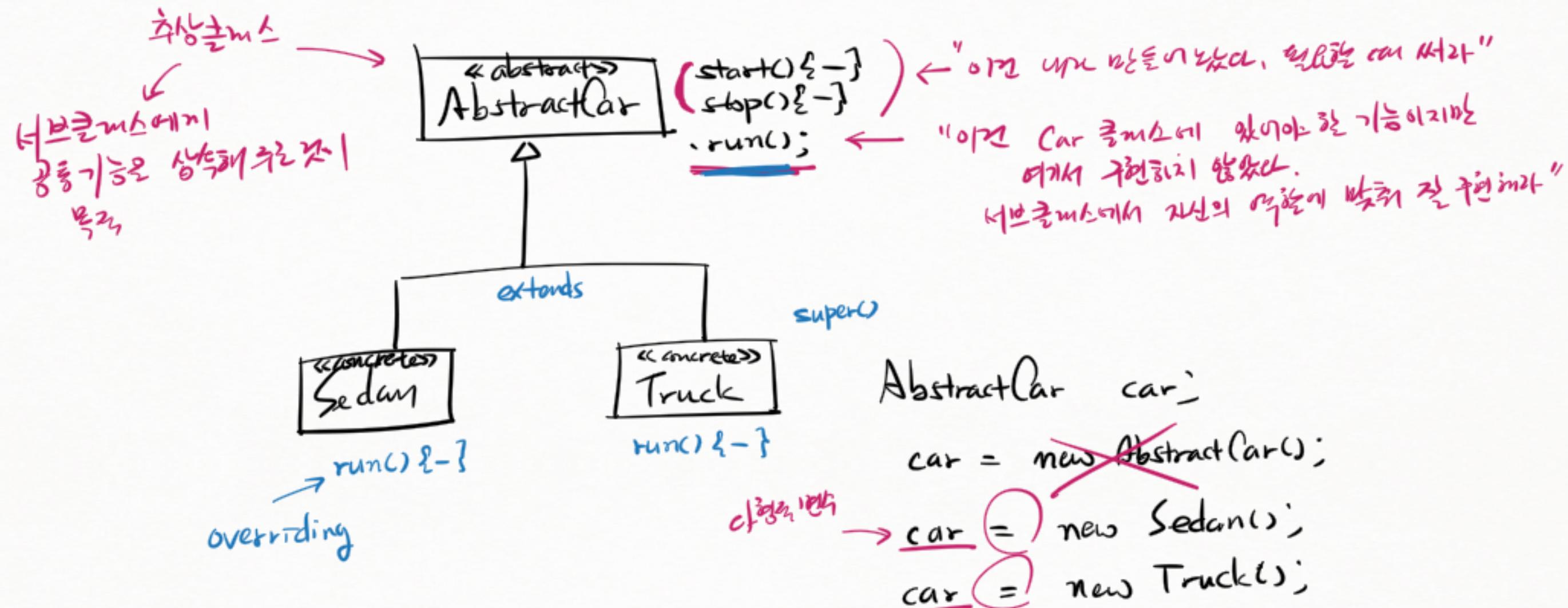


구례여러 : 각 구체에게 봄이로 선택 가능

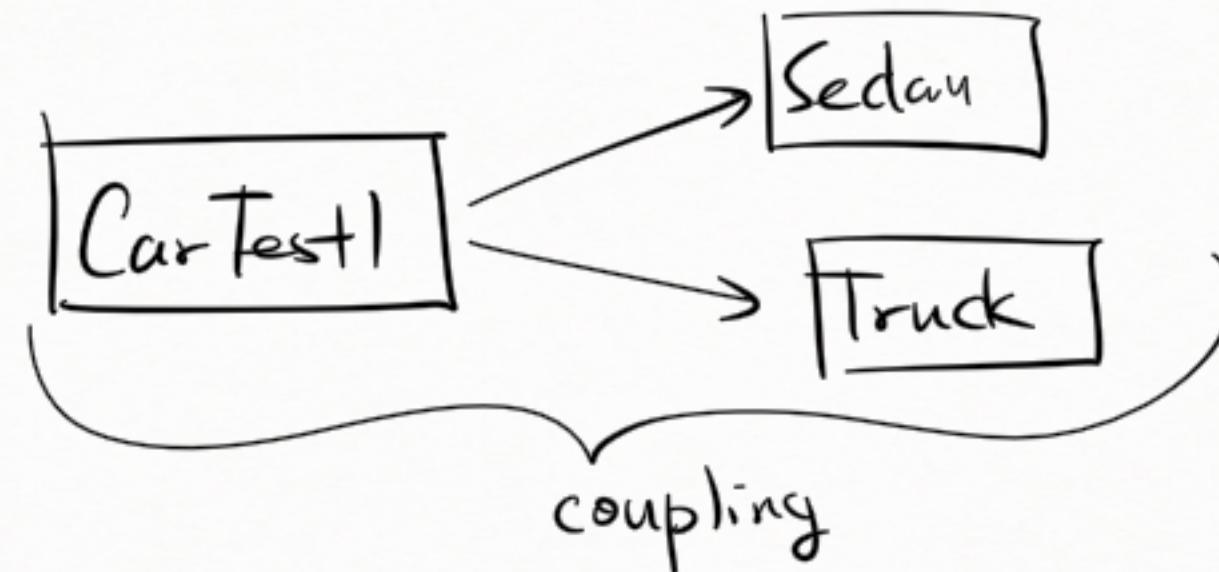
Nas 8m3 (2km)

기 32
Korean 03
한국어 03
03d.

* 추상클래스와 구현



* 실전 프로그래밍 1 단계



coupling

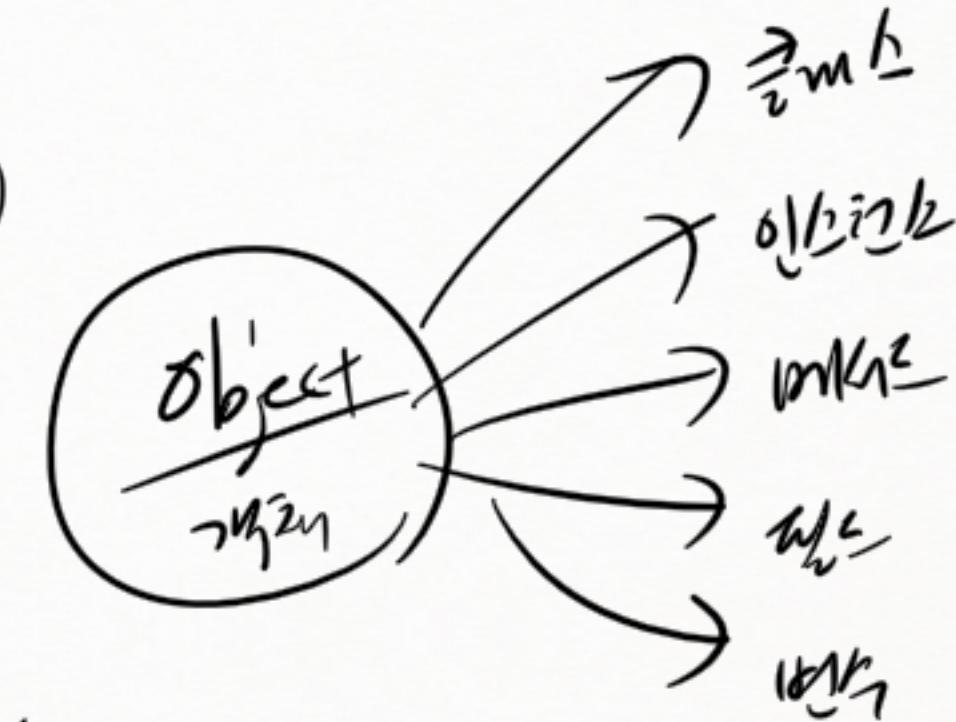
- openedSunroof
- auto
- **cc value**
- start() { - }
- stop() { - }
- ✓ · run() { - }
- openSunroof() { - }
- closeSunroof() { - }

Sedan

Truck

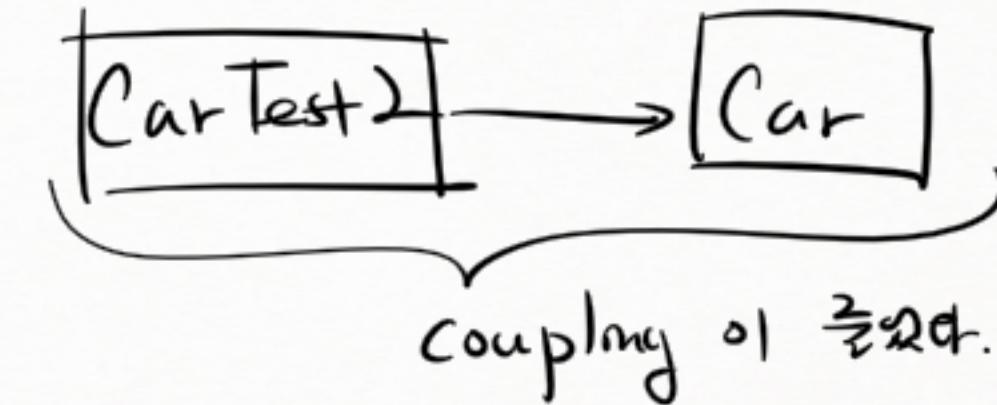
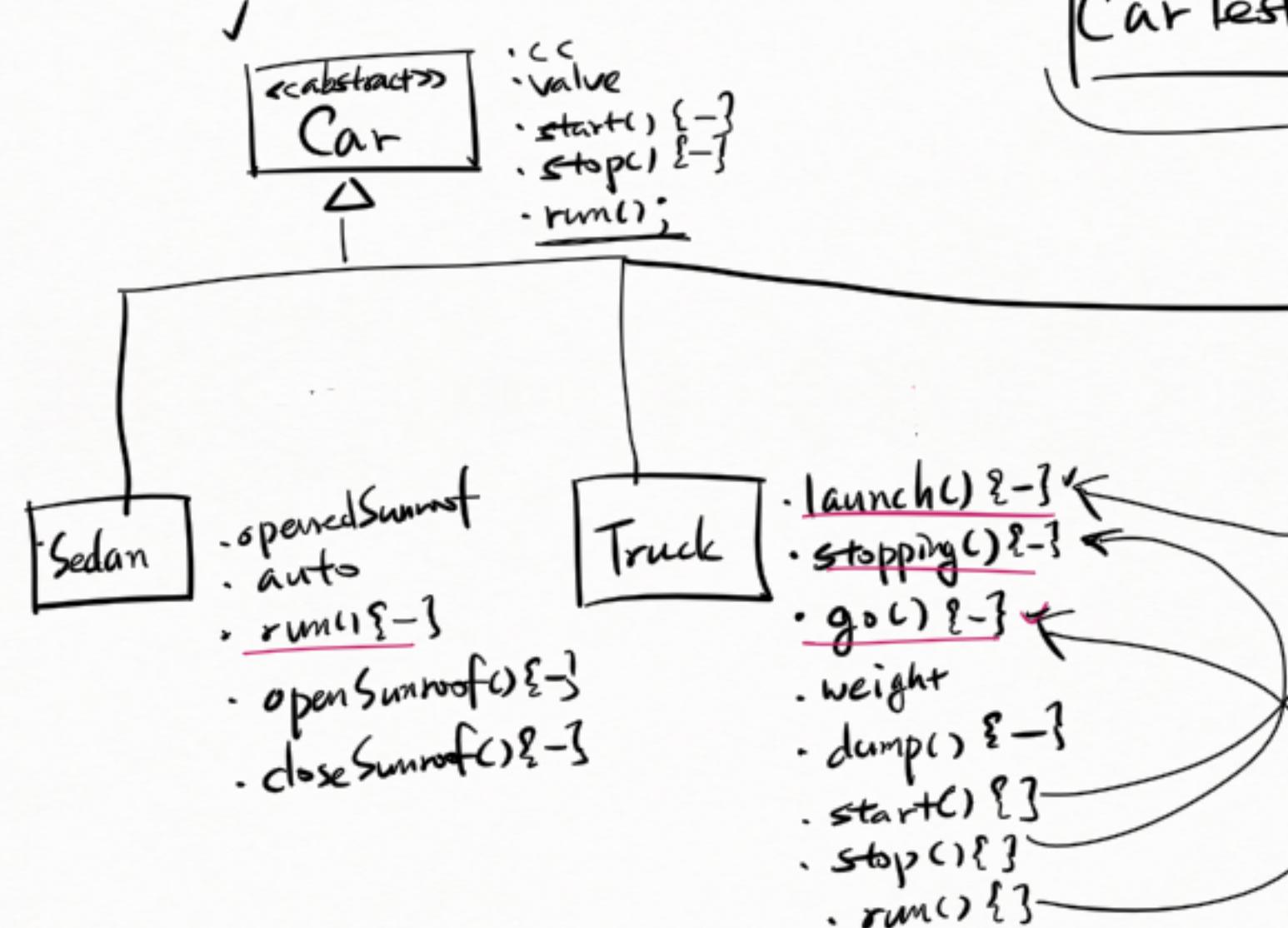
Truck

- **cc value**
- launch() { - } ✓
- stopping() { - } -
- go() { - } ✓
- weight
- dump() { - }



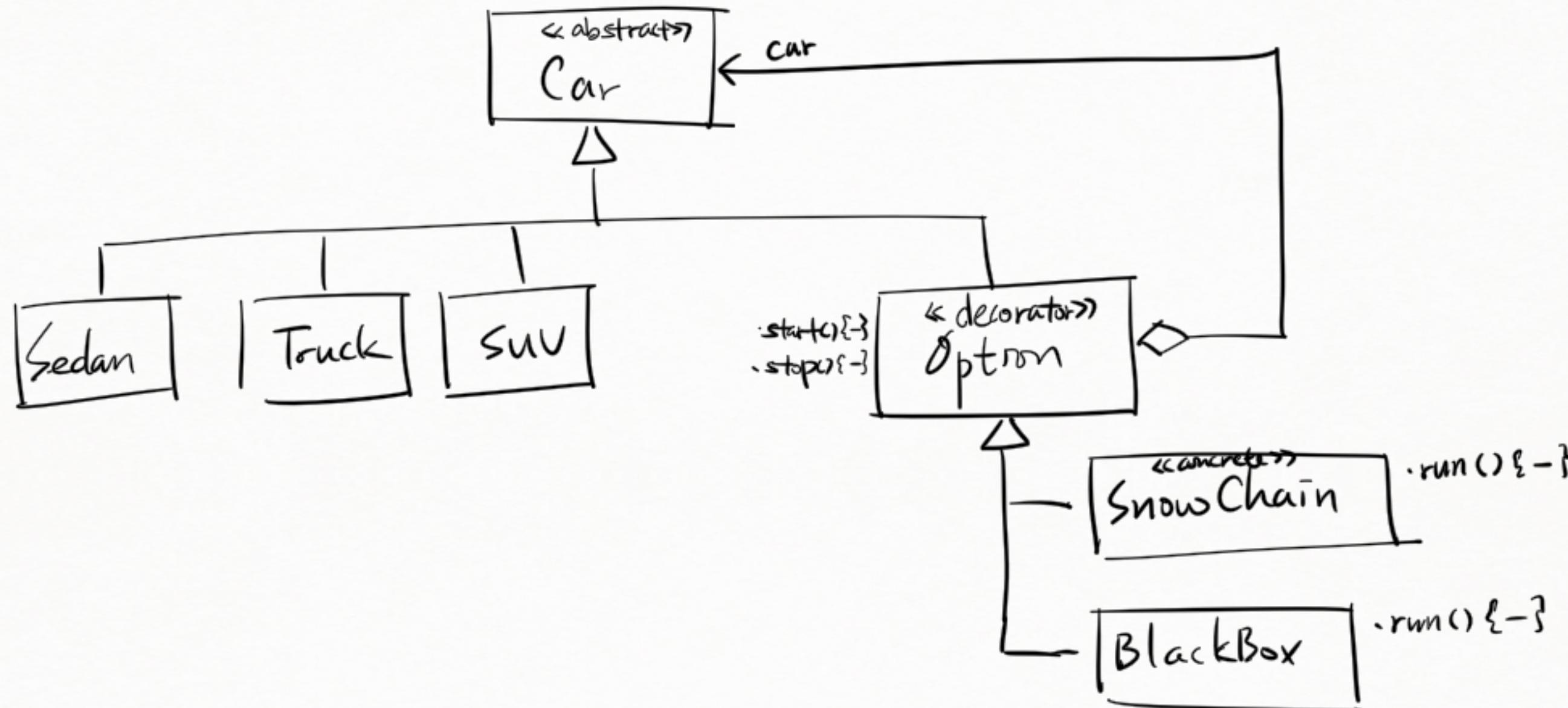
* 상 하 한 한 한 한 한 - 품종, 품질, 성능

\hookrightarrow : generalization

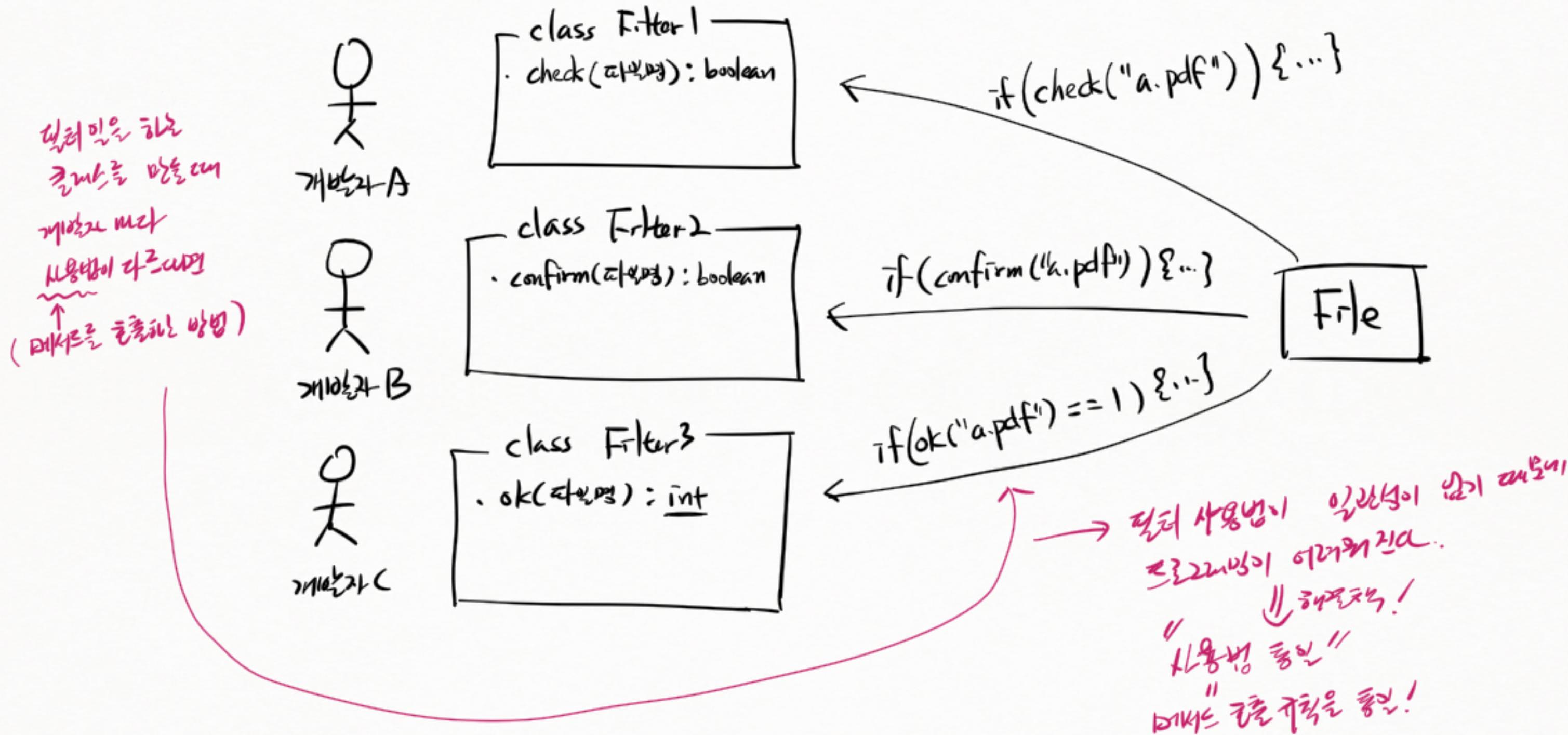


- enableLwd: boolean
- activateLwd(boolean)
- run() {-}

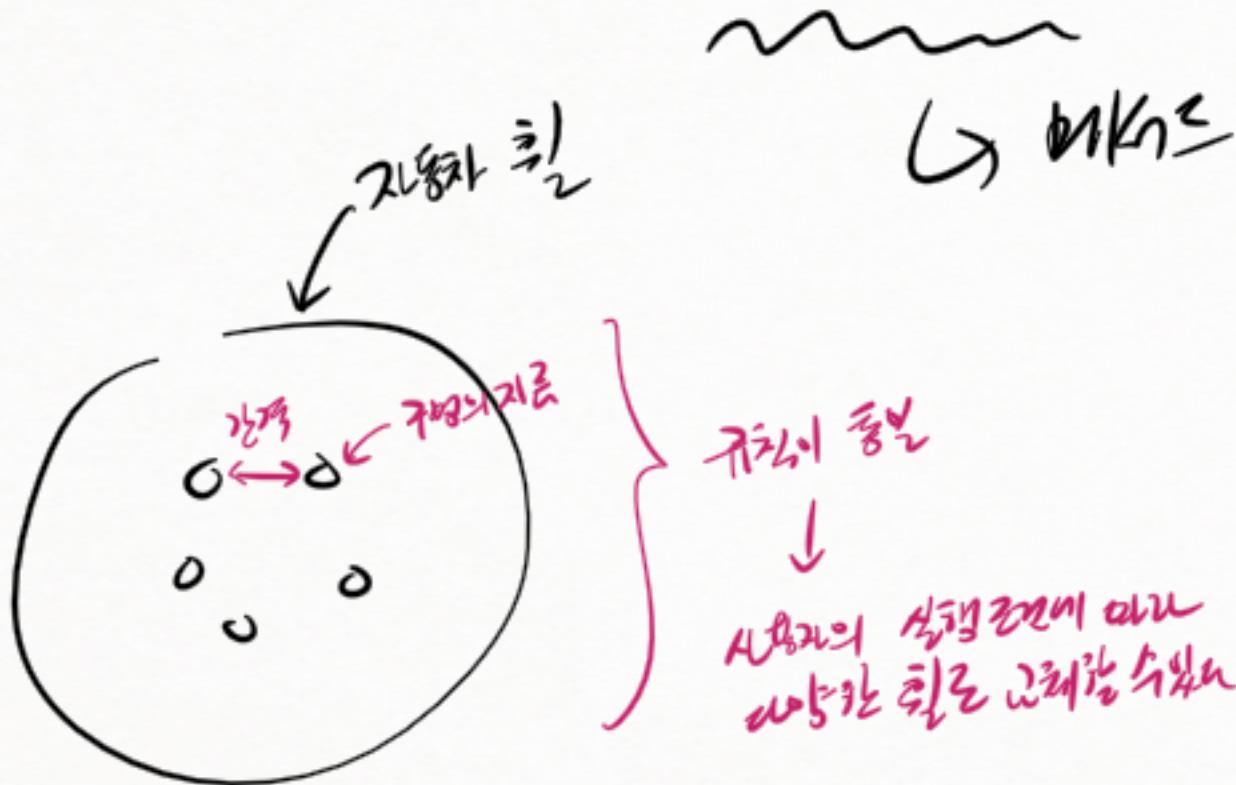
* 차선을 3개로 나누면 차량이 1개로 통합



* 파일 검색과 필터의 관계



* 인터페이스



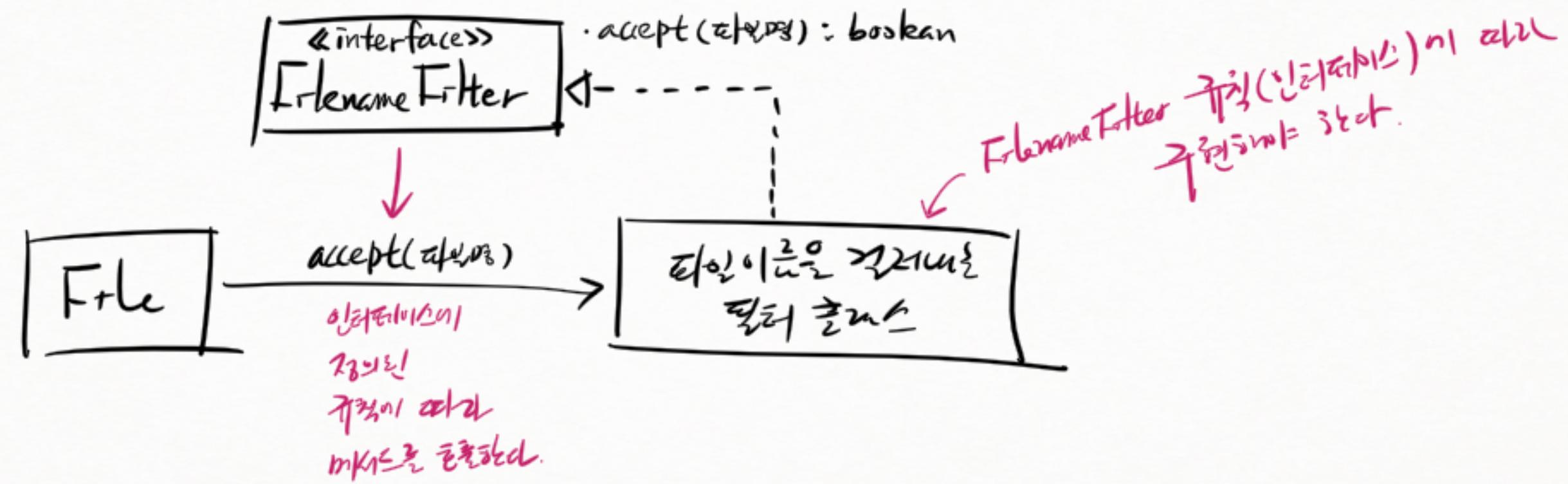
↳ Mammal은 Animal을 구조로 상속하는 외형

- Mammal은 Animal로 사용이 상관없이
동일한 시그니처(METHOD, 멤버변수, 리턴타입)를 갖는 Mammal을 구현한다.

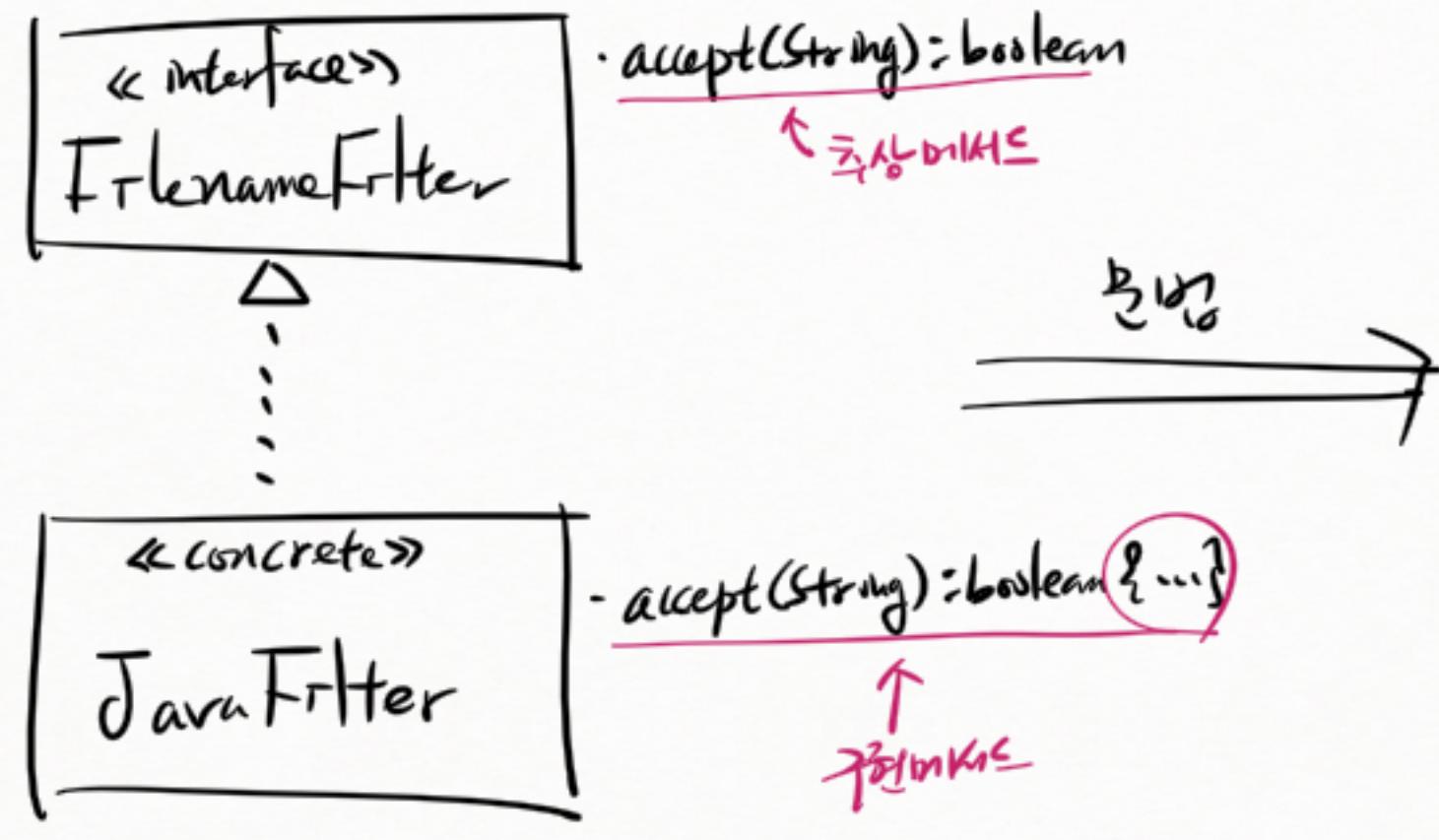
제작자는 개발자가 상관없이
일관된 방식으로
사용할 수 있다.
↓
Mammal 구조로 재구성 가능!

제작자는 개발자가 상관없이
일관된 방식으로 사용할 수 있다.
즉 제작자는
구조의 고체화가 가능하다.

* 자료 필터와 인터페이스

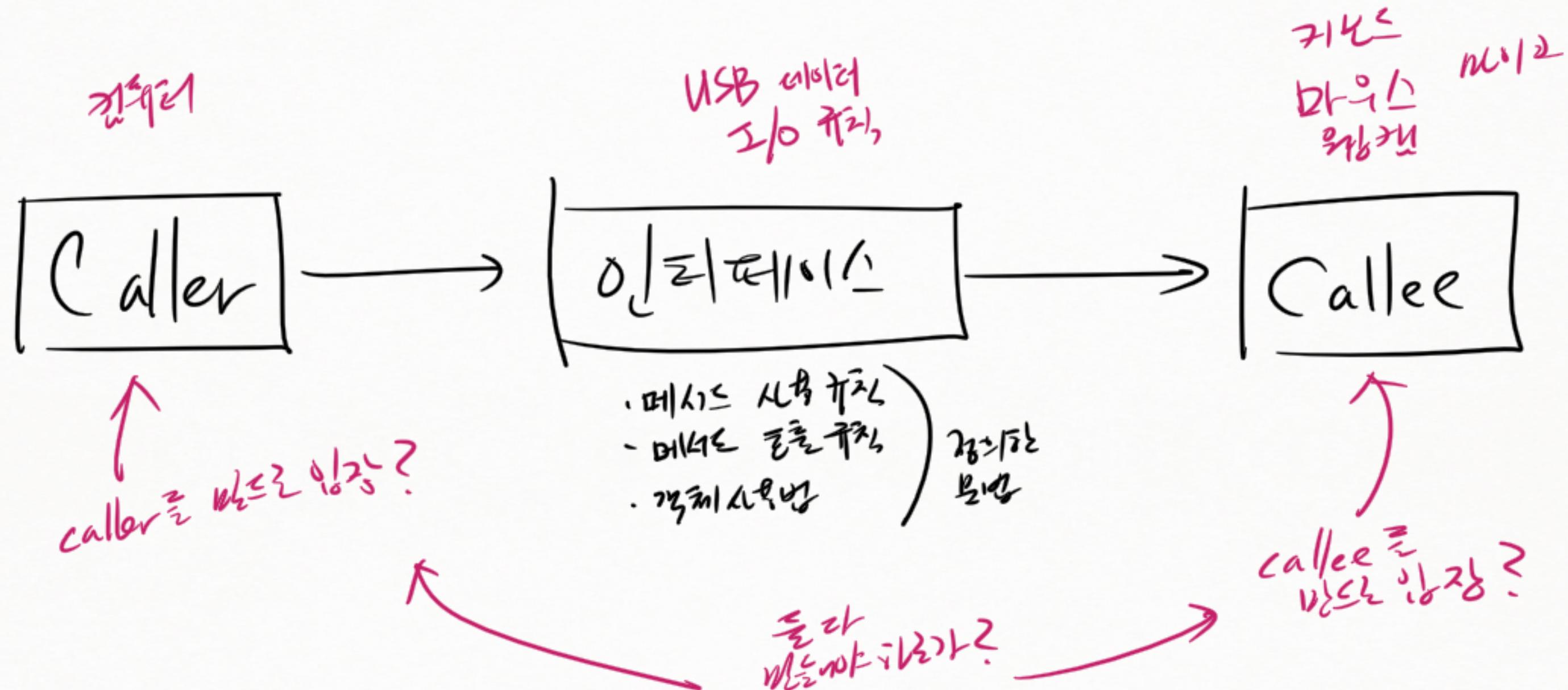


* FilenameFilter 인터페이스 구현하기



```
class JavaFilter  
    implements FilenameFilter  
  
    public boolean accept(Storm name)  
    {  
        ==  
    }  
  
}  
  
}  
  
↑  
Filenam Filter  
nächste  
Zeile nicht  
erlaubt!
```

* OOP 인터페이스



* FilenameFilter 인터페이스

