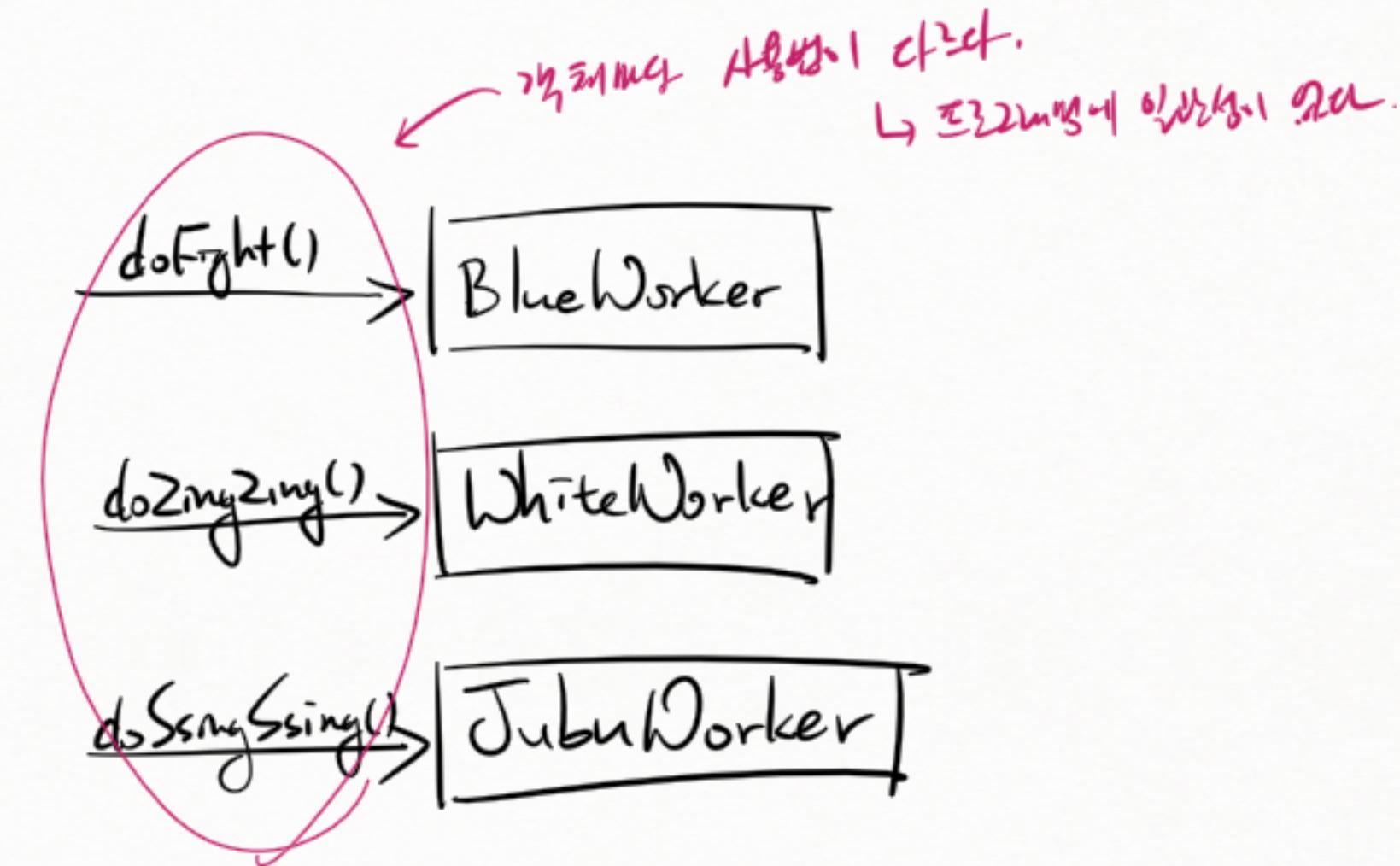
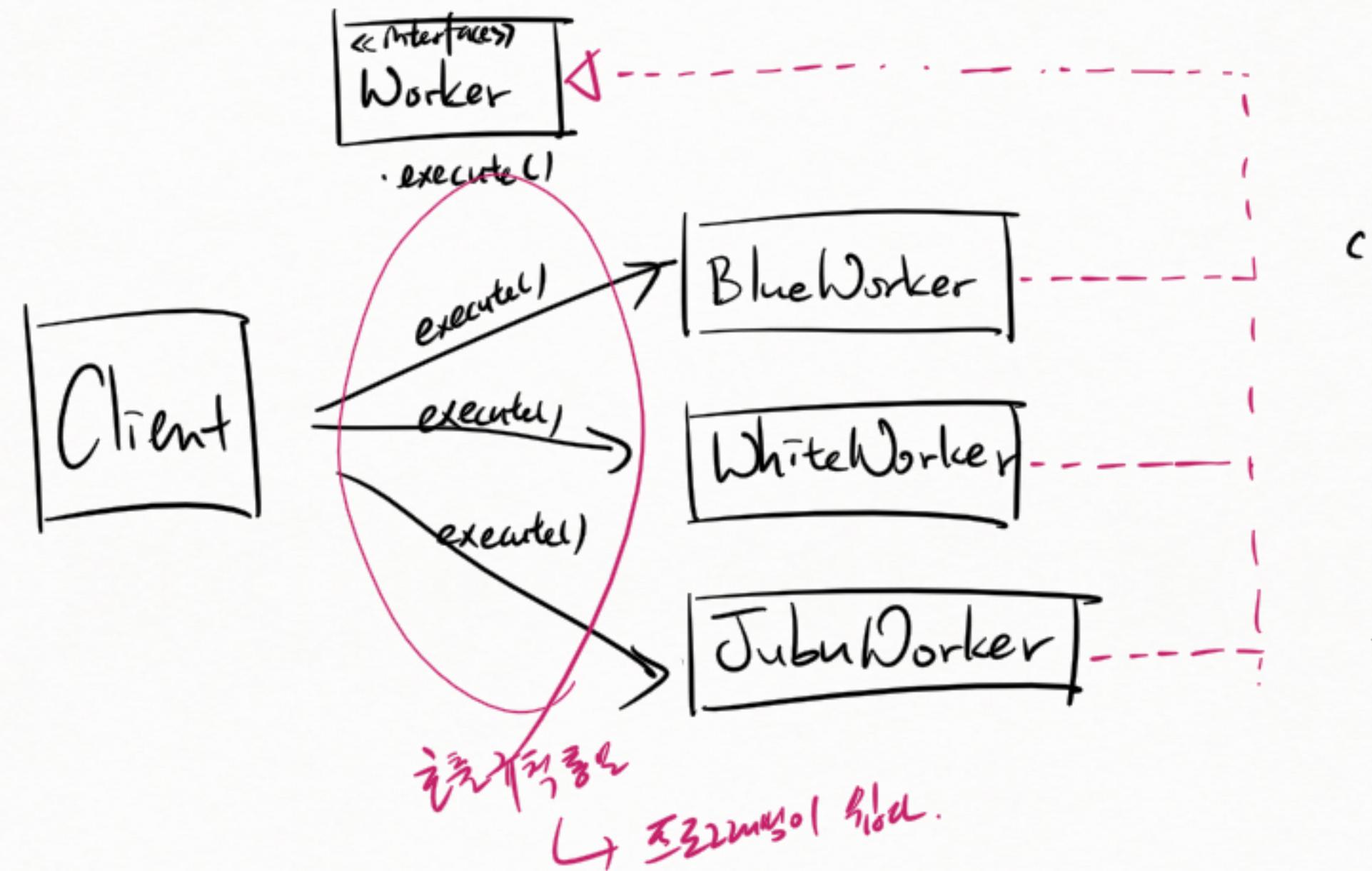


인터페이스 (Interface)

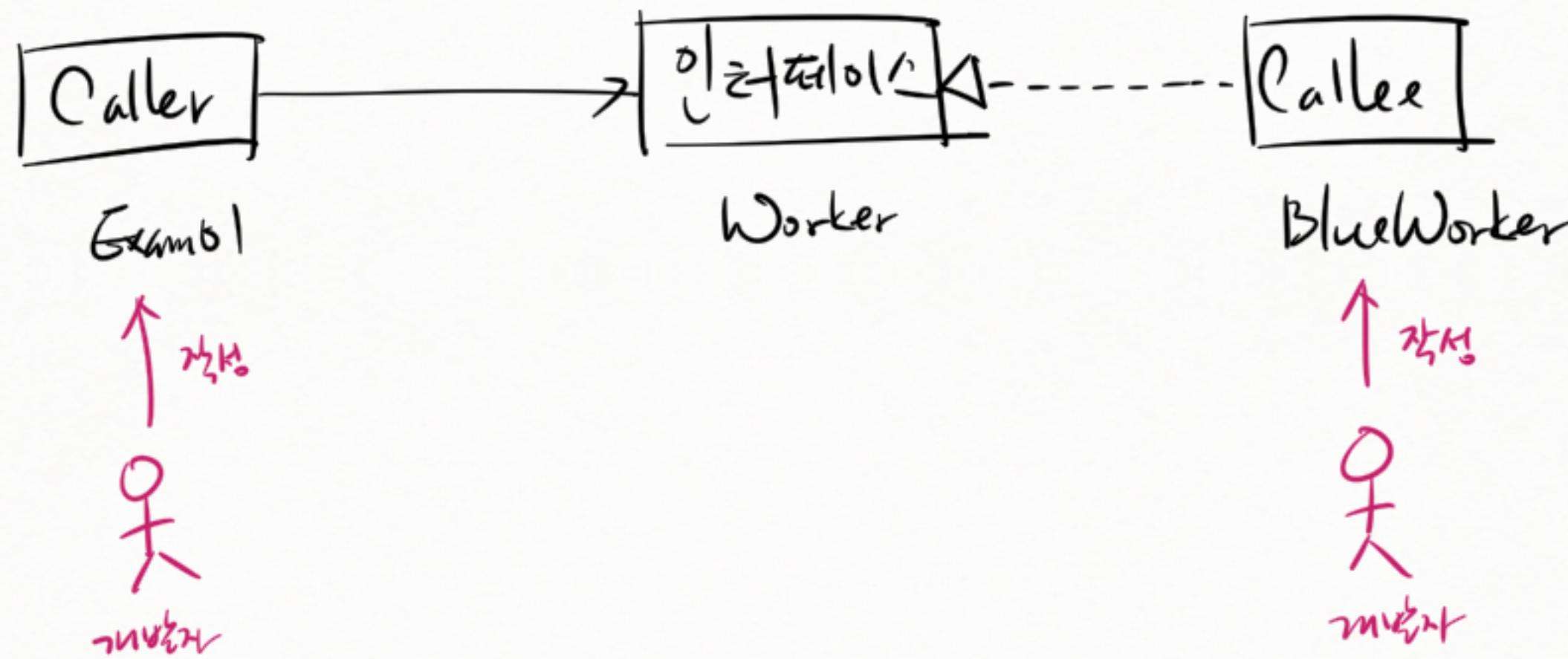
## \* 인터페이스 사용 전



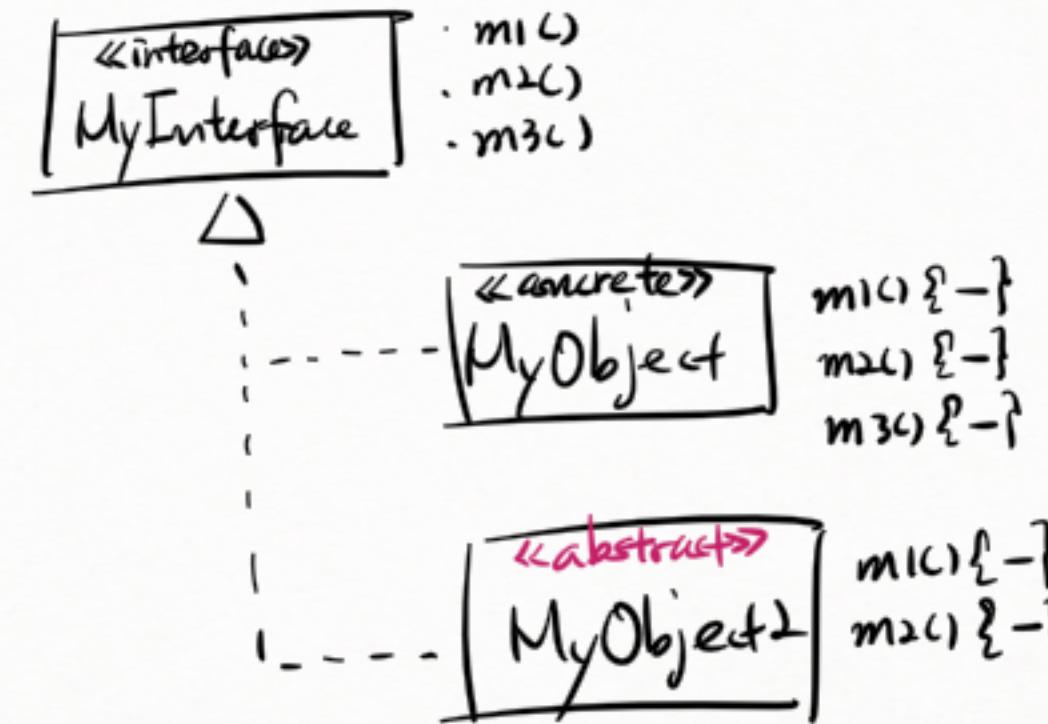
\* 인터페이스 사용 후



\* 인터페이스 와 caller / callee



## \* 인터페이스의 구현



MyInterface ref;

ref = new MyObject();

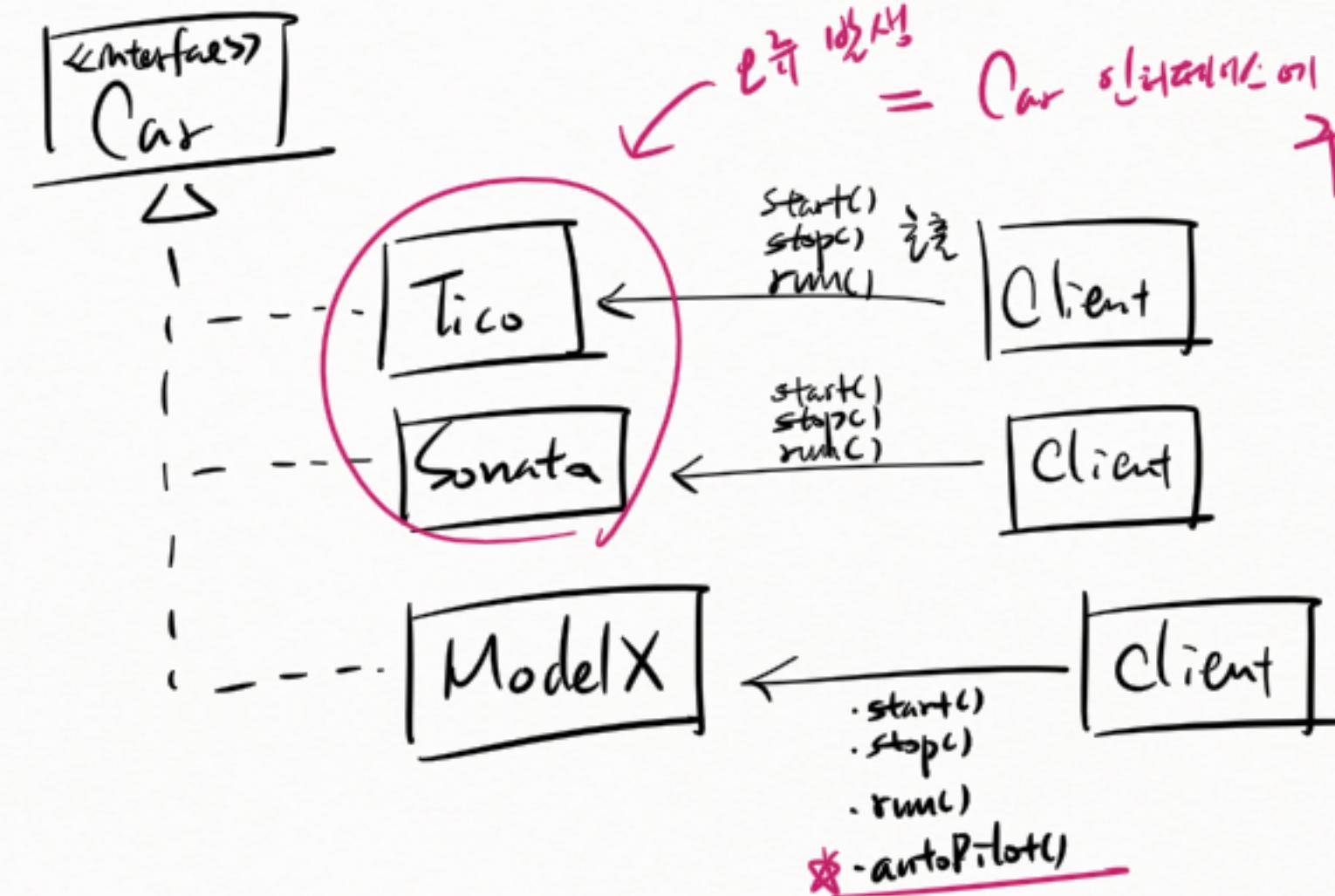


인터페이스를 구현한 클래스의  
인스턴스화

\* default 메서드 사용

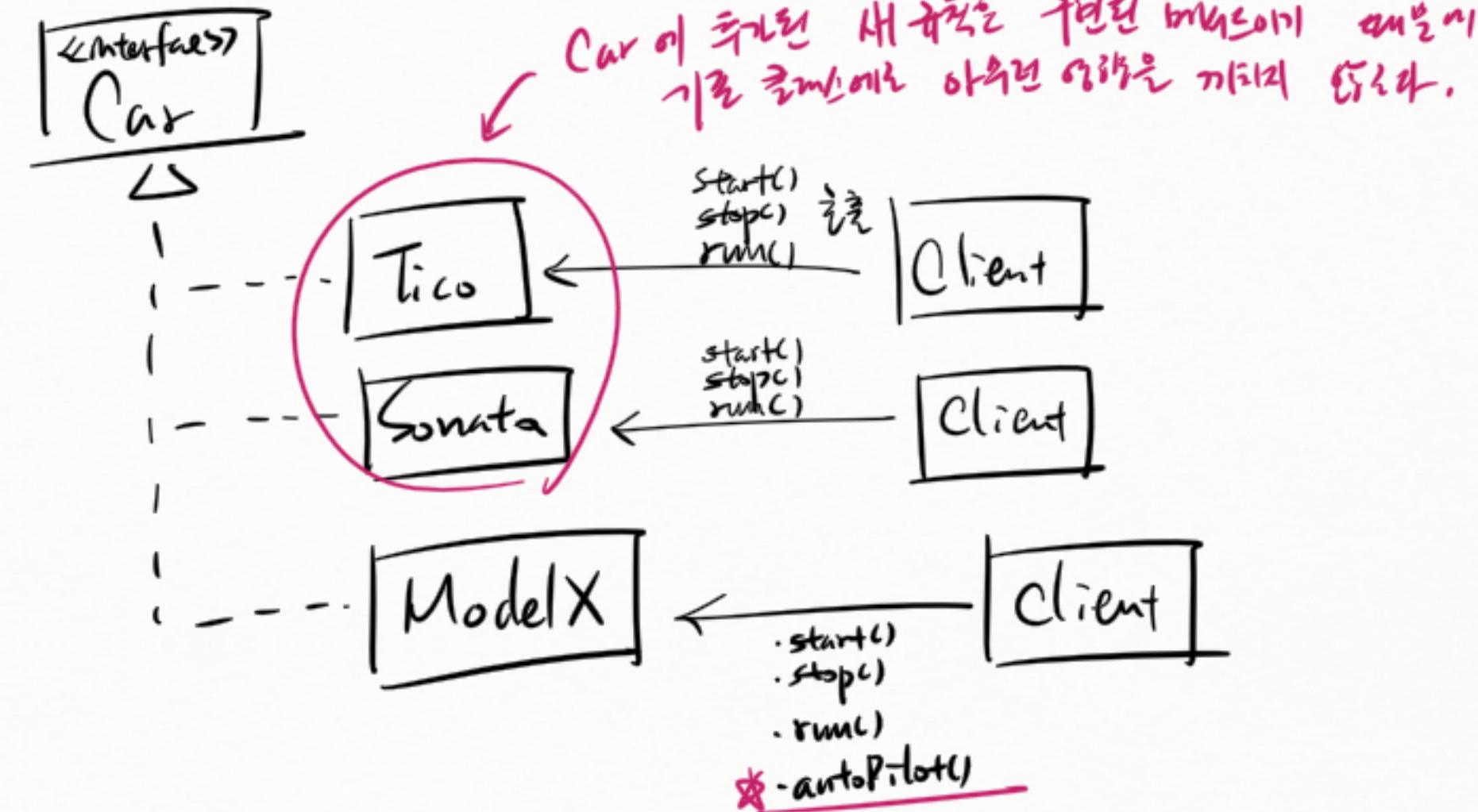
```
interface Car {  
    - start();  
    - stop();  
    - run();  
}
```

↑  
구직 추가  
+ autopilot();

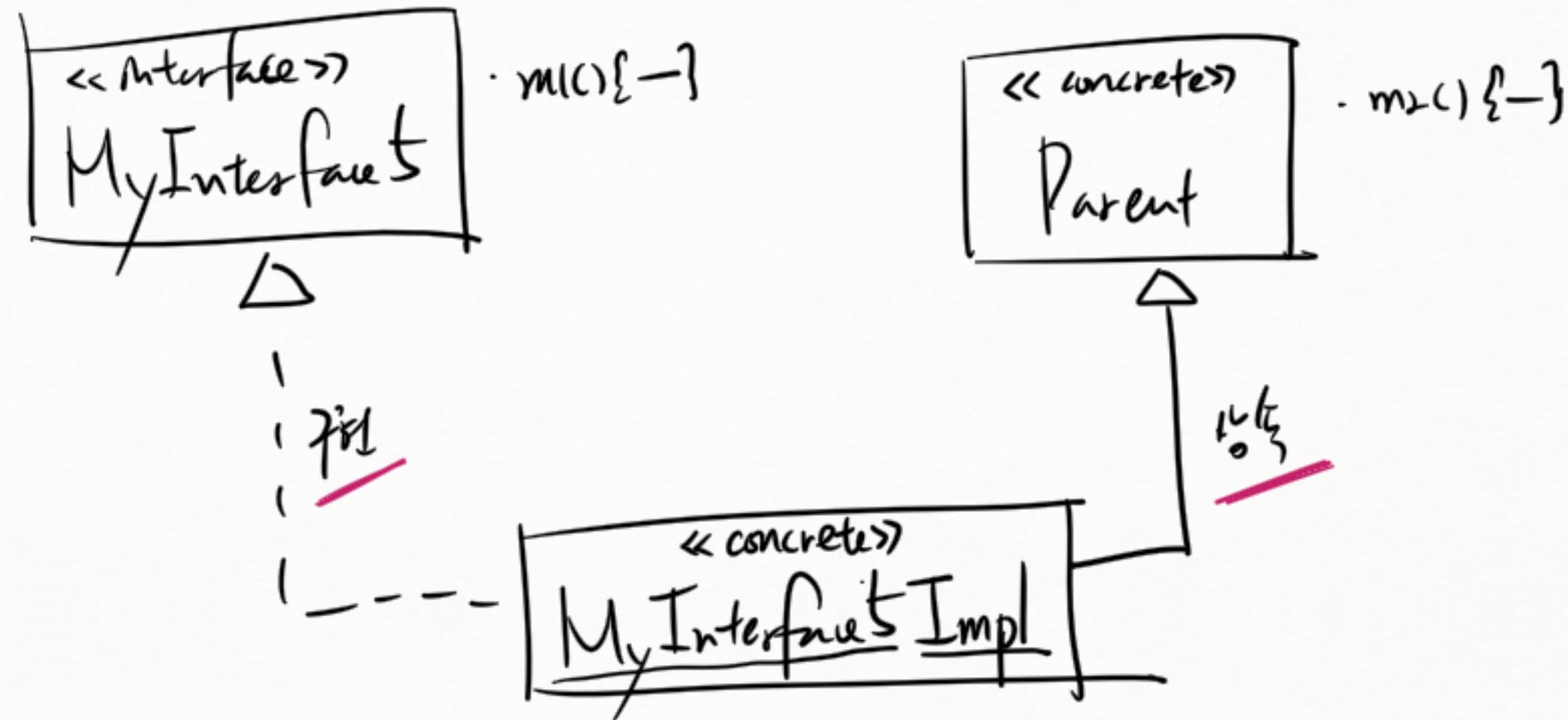


\* default interface 헬퍼

```
interface Car {  
    - start();  
    - stop();  
    - run();  
}  
  
default autopilot() {}
```

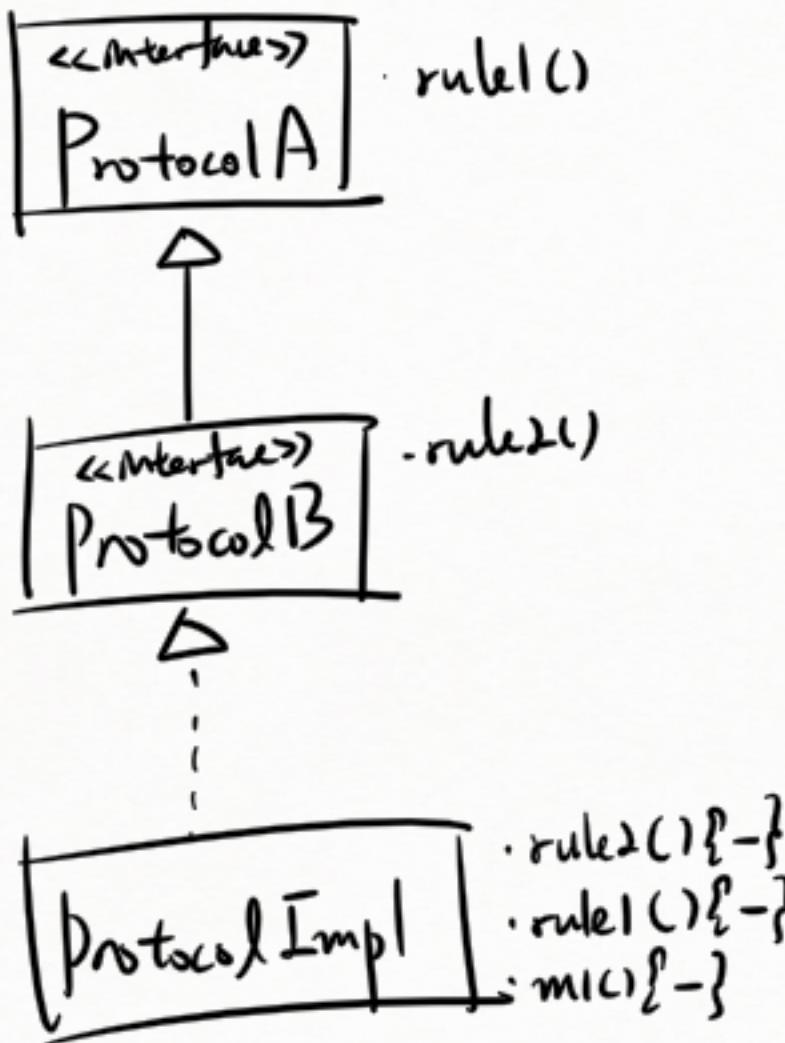


\* static b1/b2



\* 인터페이스 상속 구현

↳ 인터페이스를 통한 메서드 호출 방식



ProtocolImpl obj = new ProtocolImpl();

obj. m1();  
obj. rule1();  
obj. rule2();

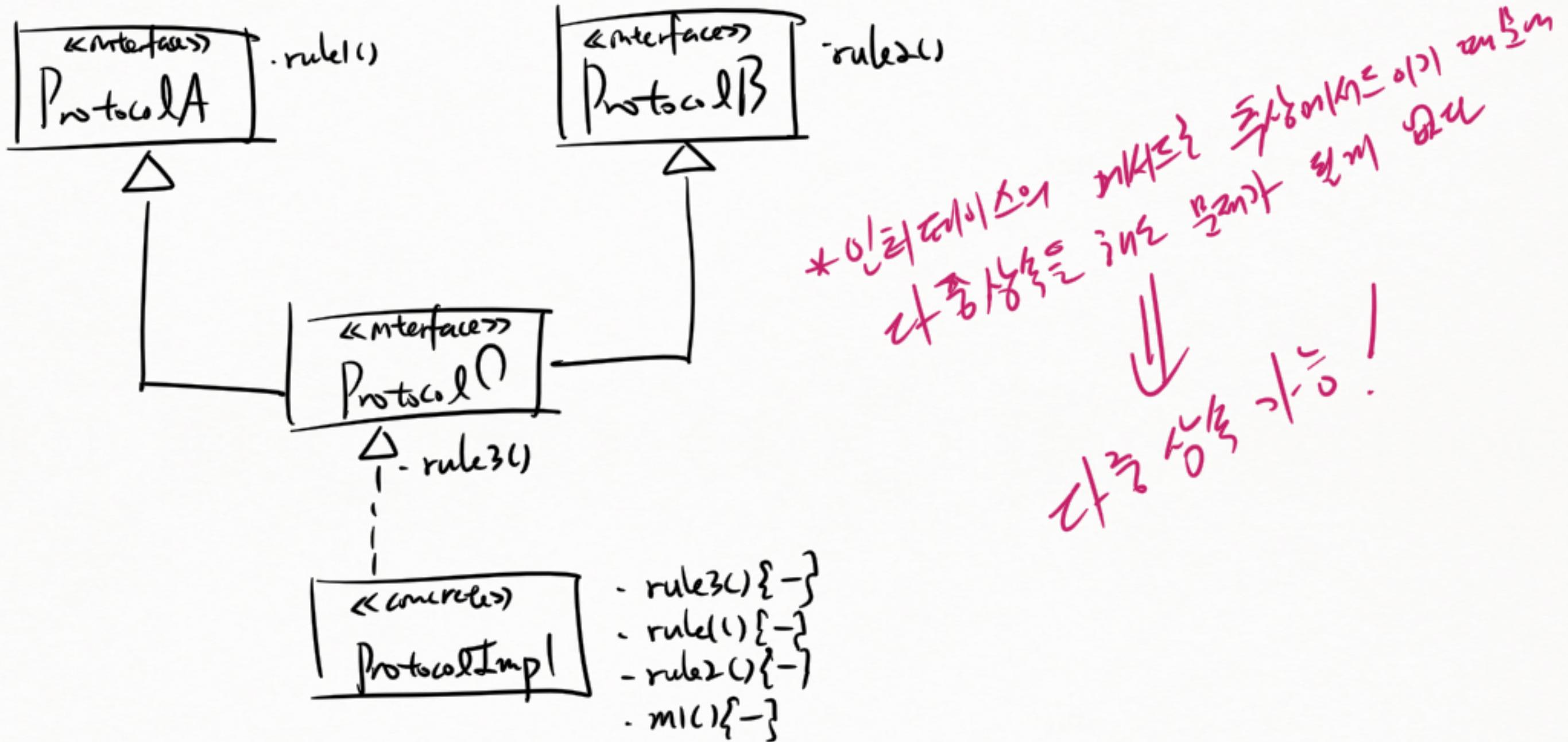
ProtocolB obj2 = obj;

~~obj2. m1();~~  
~~obj2. rule2();~~  
~~obj2. rule1();~~

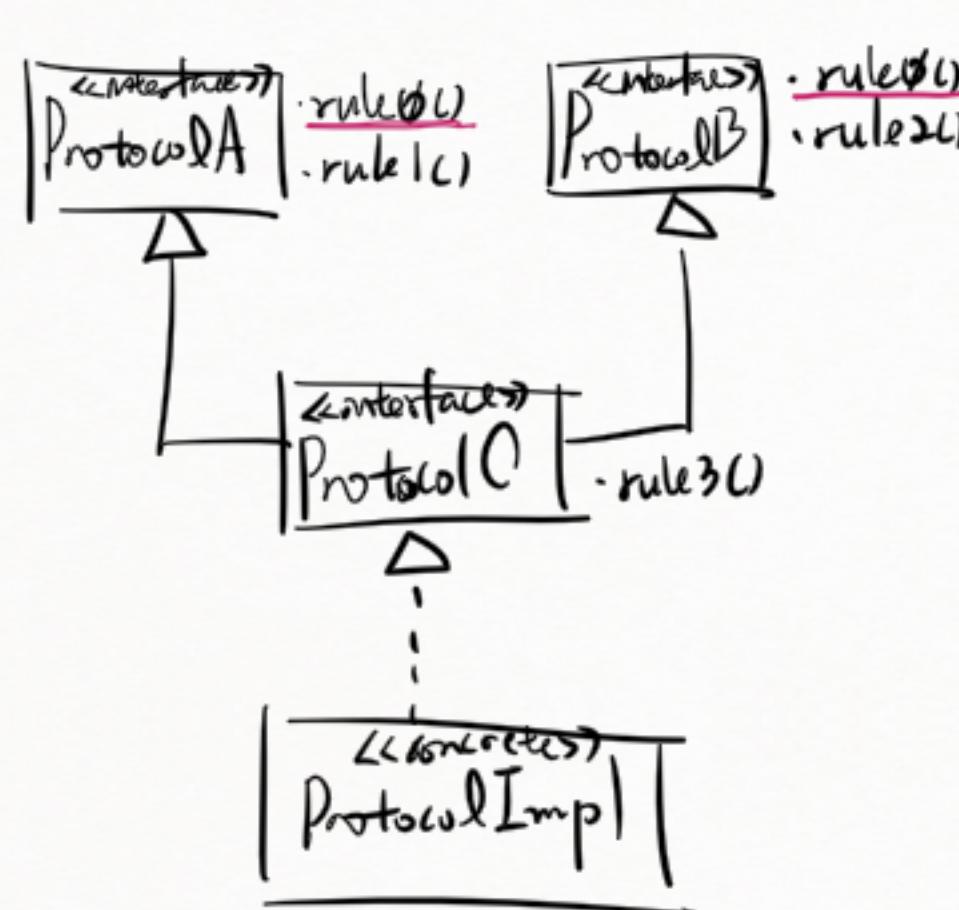
~~obj3. m1();~~  
~~obj3. rule2();~~  
~~obj3. rule1();~~

ProtocolA obj3 = obj;

\* 인터페이스 다중 상속



\* 이전에는 다음과 같은  
가능한 경우

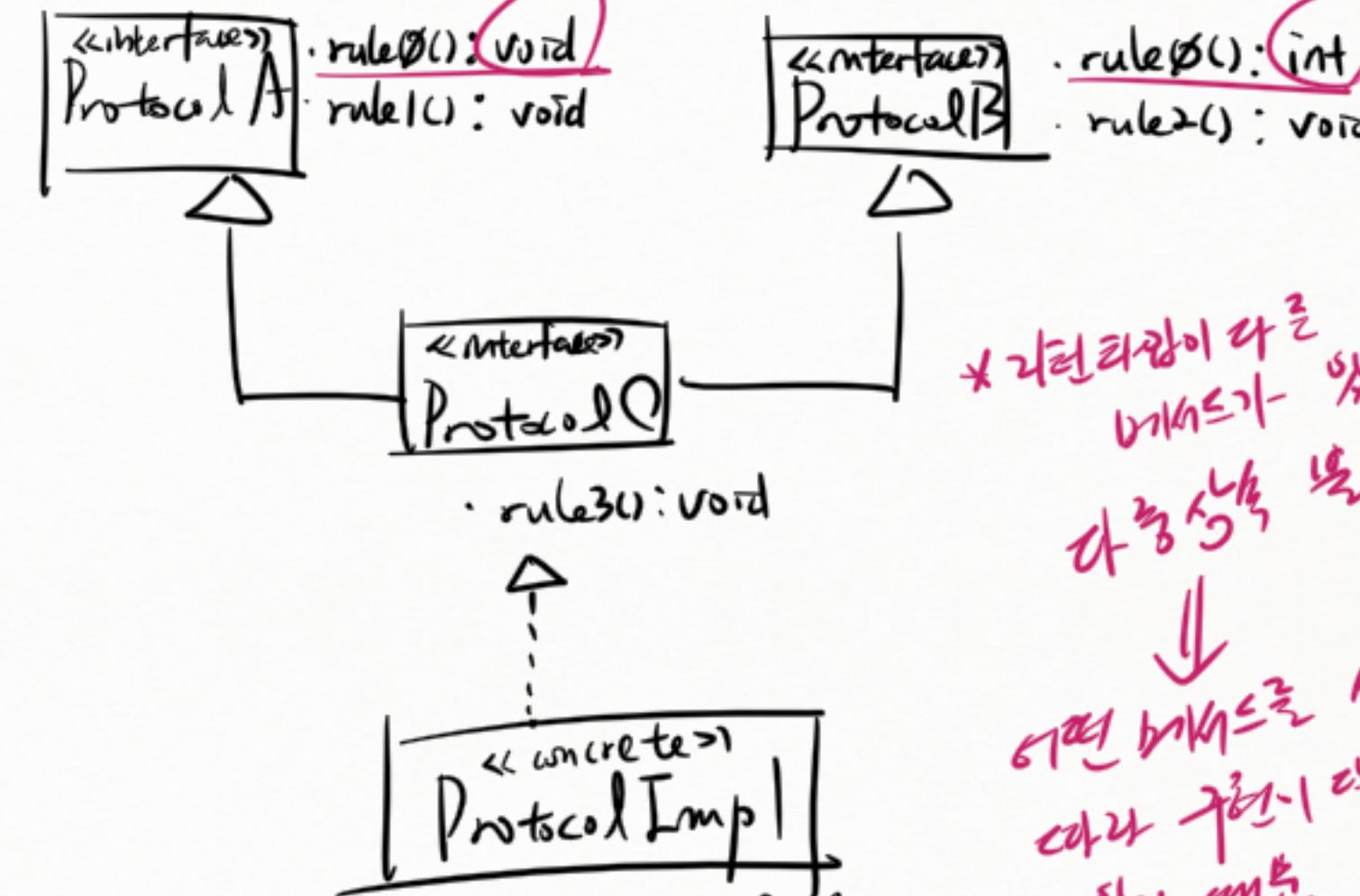


ProtocolA  
rule0()  
rule1()

ProtocolB  
rule0()  
rule2()

ProtocolC  
rule3()

불가능한 경우



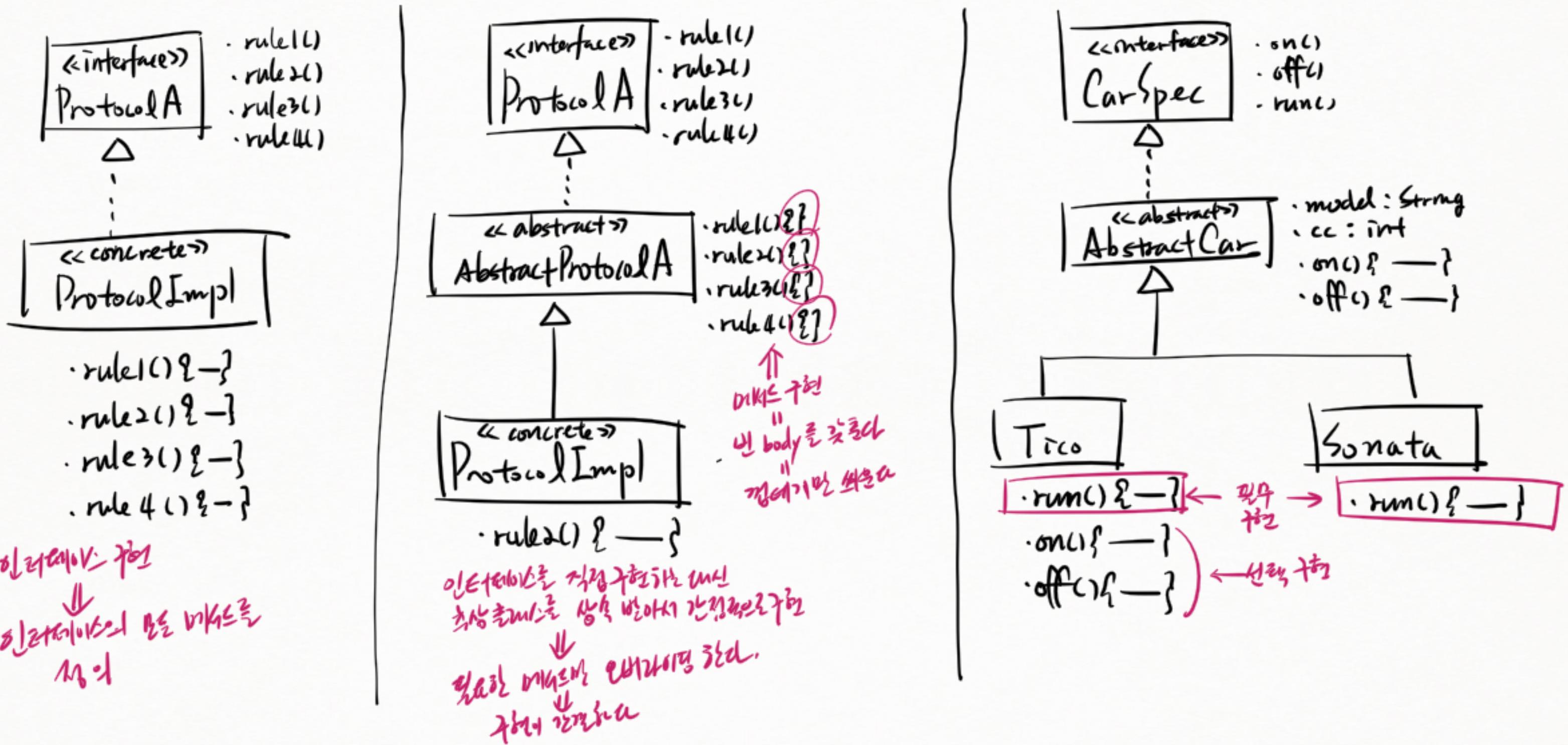
이전처럼 다를  
수는 없지만  
부모가 다른  
것은 가능합니다.

• void rule0()  
• int rule0()

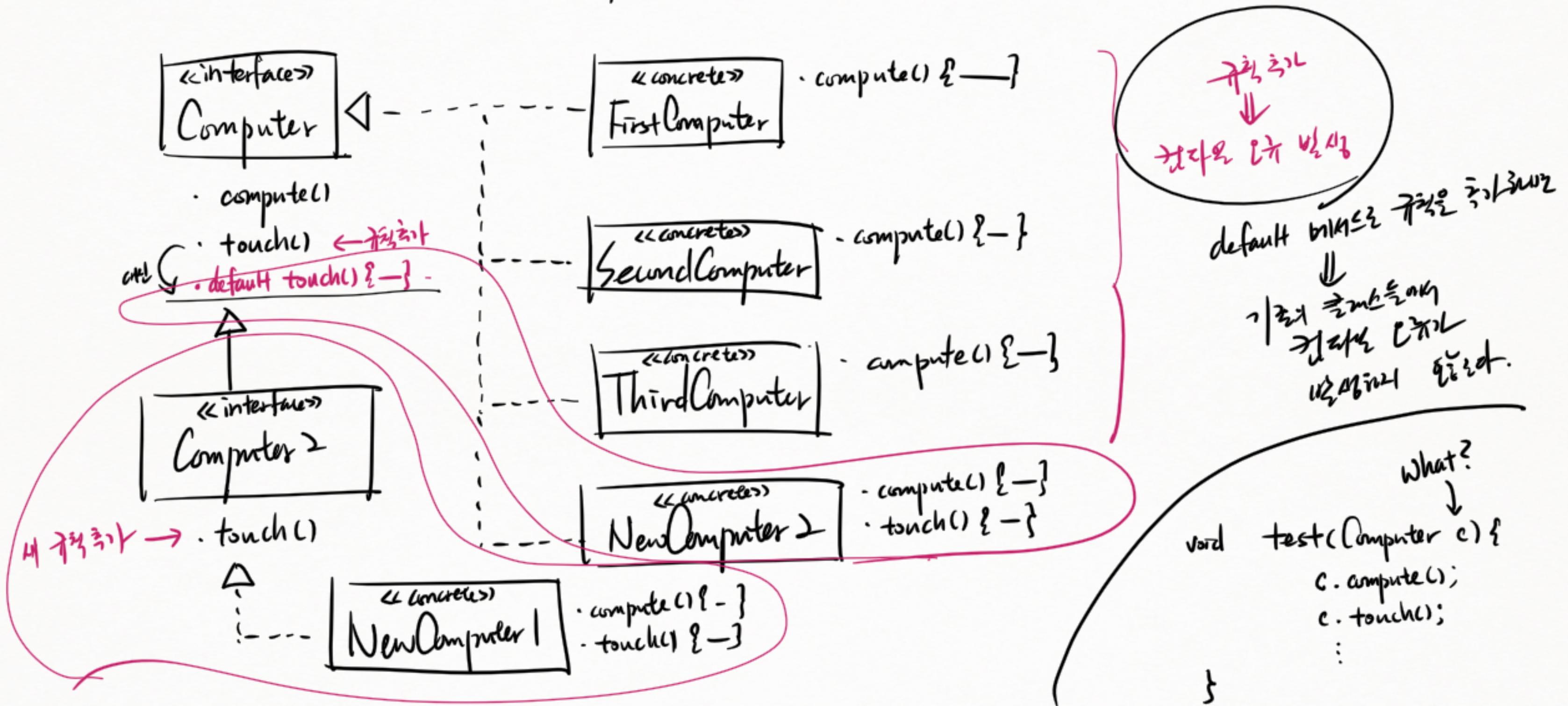
\* 이런 이유로 다를  
수는 없지만  
부모가 다른  
것은 가능합니다!

여전히 부모는 상속을  
받아 가질 수 있는  
부모

## \* 인터페이스와 추상 클래스



\* 인터페이스의 default 메소드

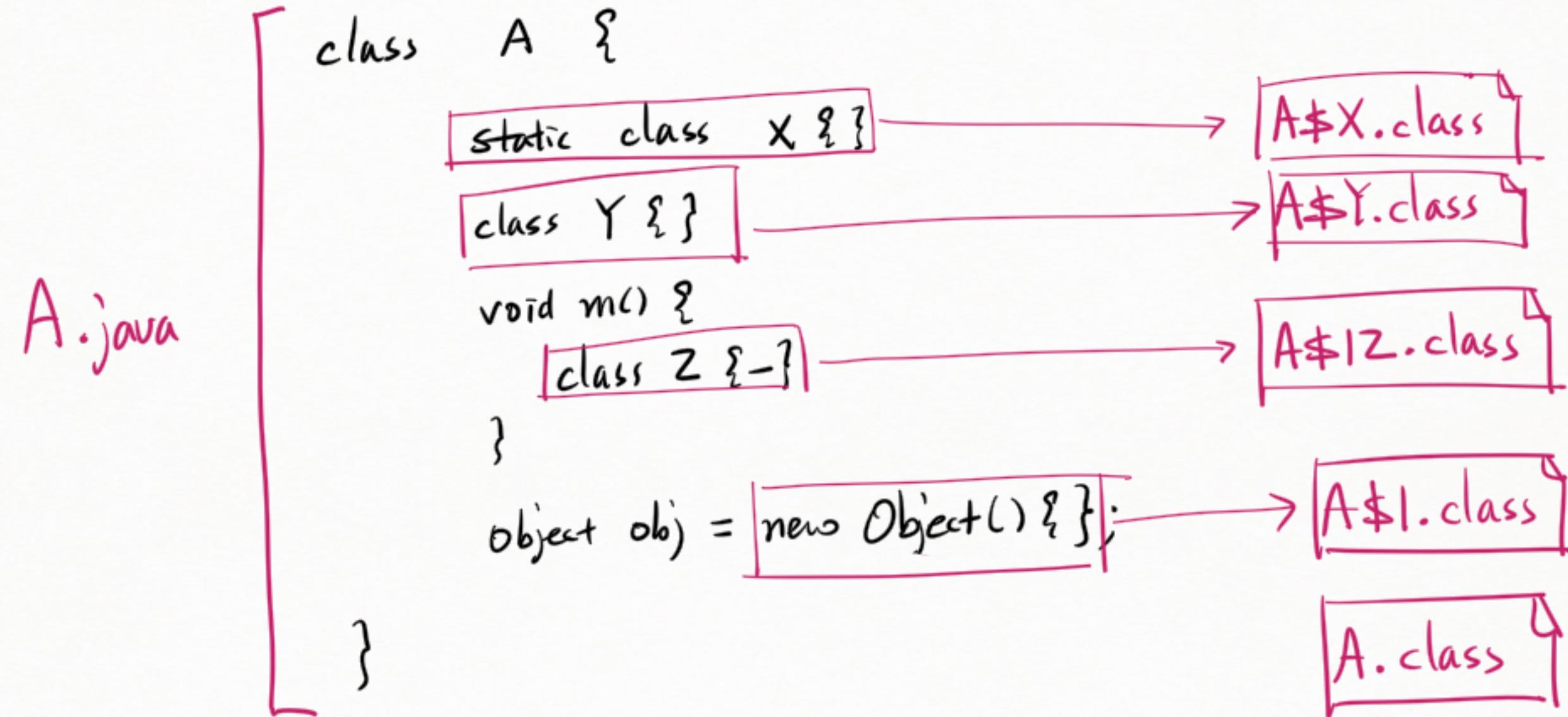


깊은  
정적  
클래스  
( nested class )

\* 중첩 클래스의 종류  
nested

```
class A {  
    static class X {} } ← static nested class  
    class Y {} } ← non-static nested class  
    void m() {  
        class Z {} } ← local class  
    }  
    Object obj = new Object() {}; ← anonymous class  
}
```

\* 중첩 클래스의 .class 파일



\* static nested class 와 non-static nested class의 차이

class A {

  static class X { }

  class Y { }

}

Y obj = new Y();

    ↓  
    new Y(this);

    ↳ Initialization  
    ↳ 바깥 클래스의 객체 주소  
    ↳ 초기화 코드로 사용된다

static class X {  
  X() { }  
}

class Y {

  A this\$0;

  Y(A arg) {

    this\$0 = arg;

  }

    ↳ 컨структор가 기본 생성자 추가

    ↳ 컨структор,

    ↳ 바깥 클래스의  
    ↳ 객체 주소를 받을  
    ↳ 레퍼런스를 추가.

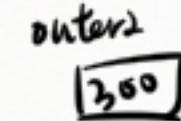
    ↳ 생성자에서  
    ↳ 바깥 클래스의 인스턴스를  
    ↳ 받을 수 있도록  
    ↳ 파라미터를 추가

B3 outer = new B3();



outer.v1 = 11;

B3 outer2 = new B3();

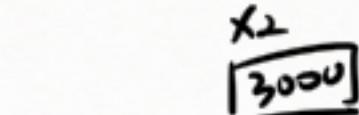


B3.X x1 = ~~outer~~.new X();



x1.test();  
↑ this

B3.X x2 =

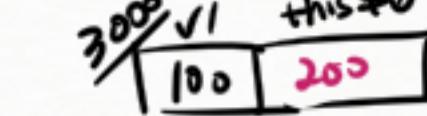


x2.test();  
↑ this

B3.X x3 =



~~outer~~.new X();



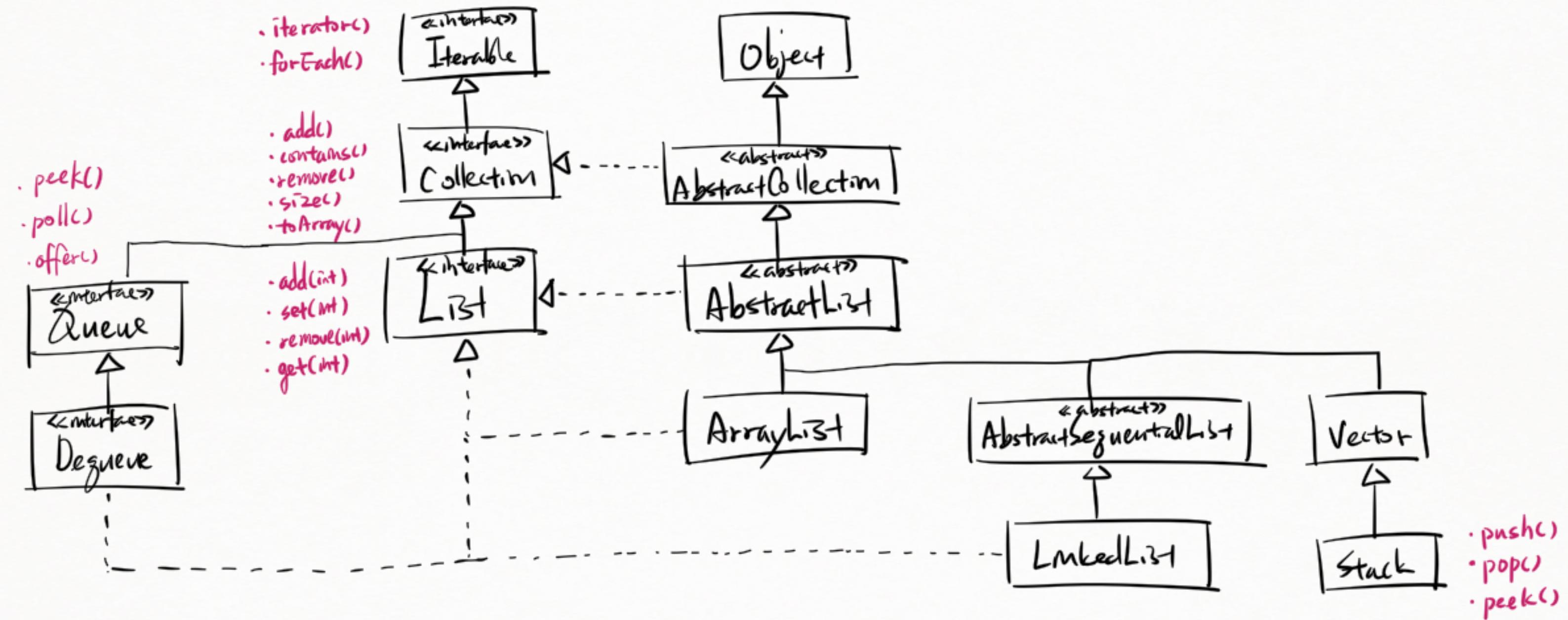
outer2.new X();



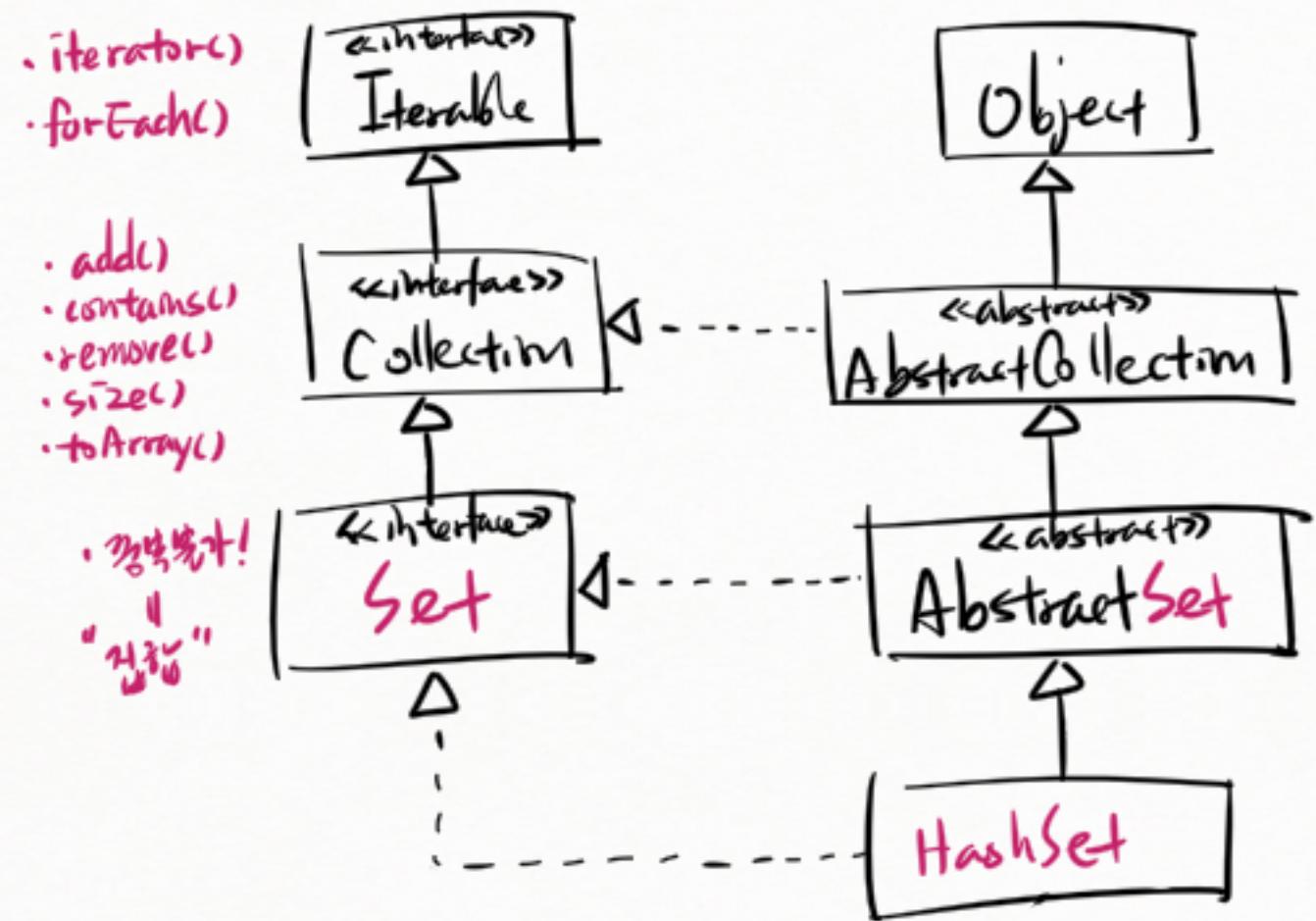
450  
x3.test();

Collection API

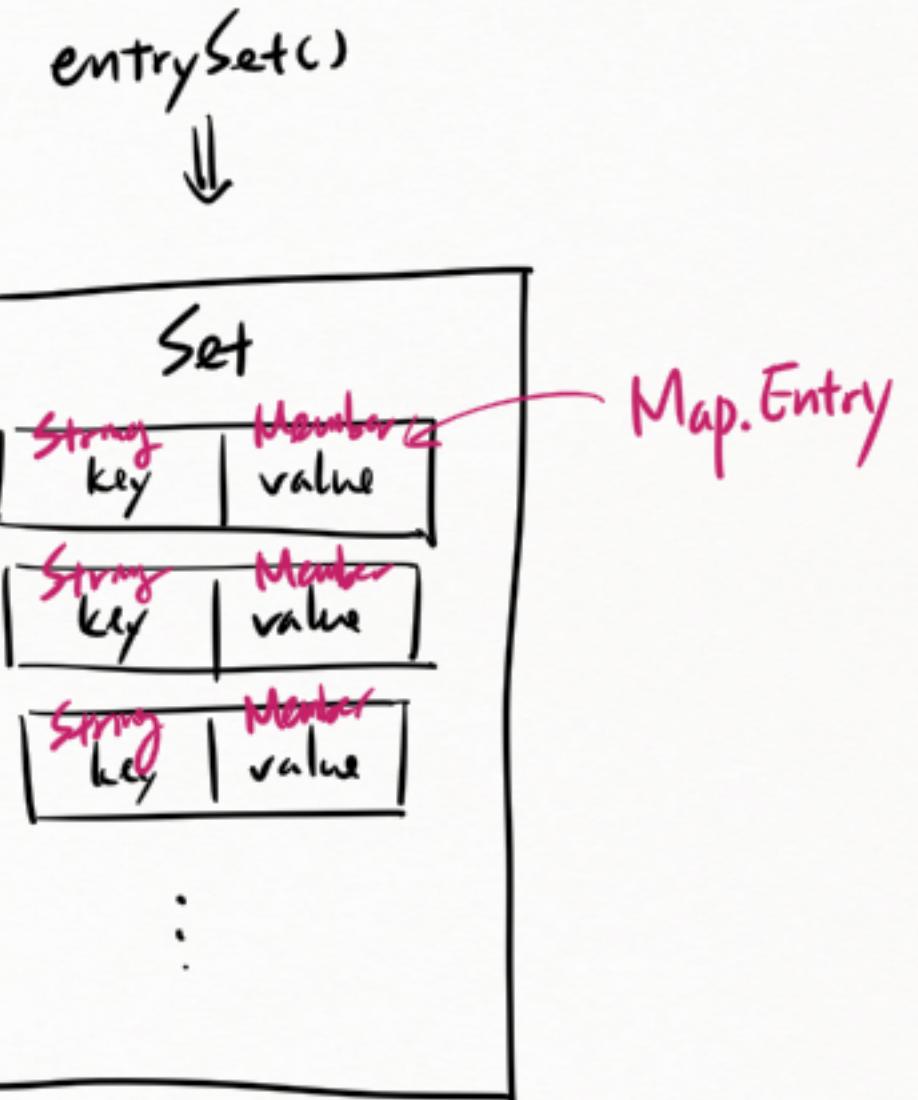
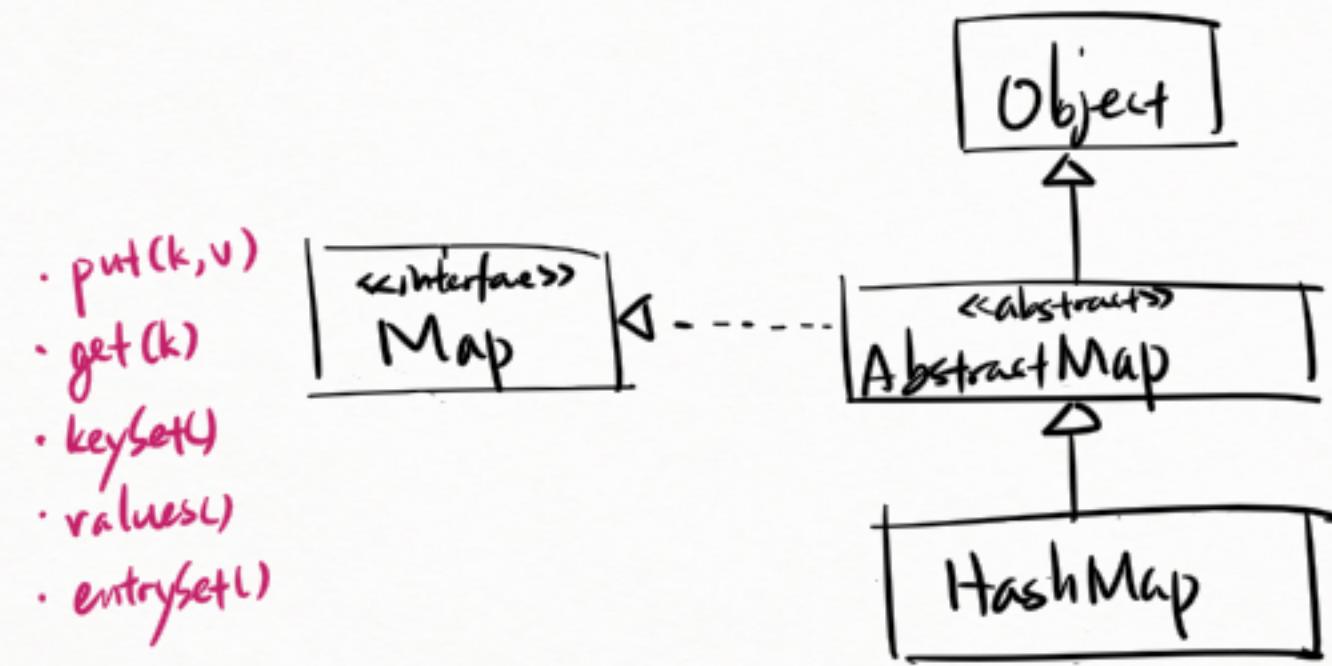
## \* List の構成と階層 (hierarchy)



## \* Set သုတေသနများ ရှိခိုင်း (hierarchy)



\* Map の構成構造 (hierarchy)



File I/O API

## \* FileOutputStream / FileInputStream

