객체 지향 프로그래밍이란?

배지영 / 09-15-23

객체 지향 프로그래밍이란?

- 객체 지향 프로그래밍은 컴퓨터 프로그래밍 패러다임 중 하나로,
  프로그래밍에서 필요한 데이터를 추상화시켜 상태와 행위를 가진 객체를 만들고
  그 객체들 간의 유기적인 상호작용을 통해 로직을 구성하는 프로그래밍 방법이다.

- 각각의 객체는 메세지를 주고받고, 데이터를 처리할 수 있다.

- 사람이 현실을 바라보는 방법을 개발에 접목
  - 직관적으로 이해하기 쉽다.
  - 유지 보수를 용이하게 만든다.

OOP를 위한 개념

- 클래스 + 인스턴스
- 추상화
- 캡슐화
- 상속
- 다형성

클래스 + 인스턴스 (객체)

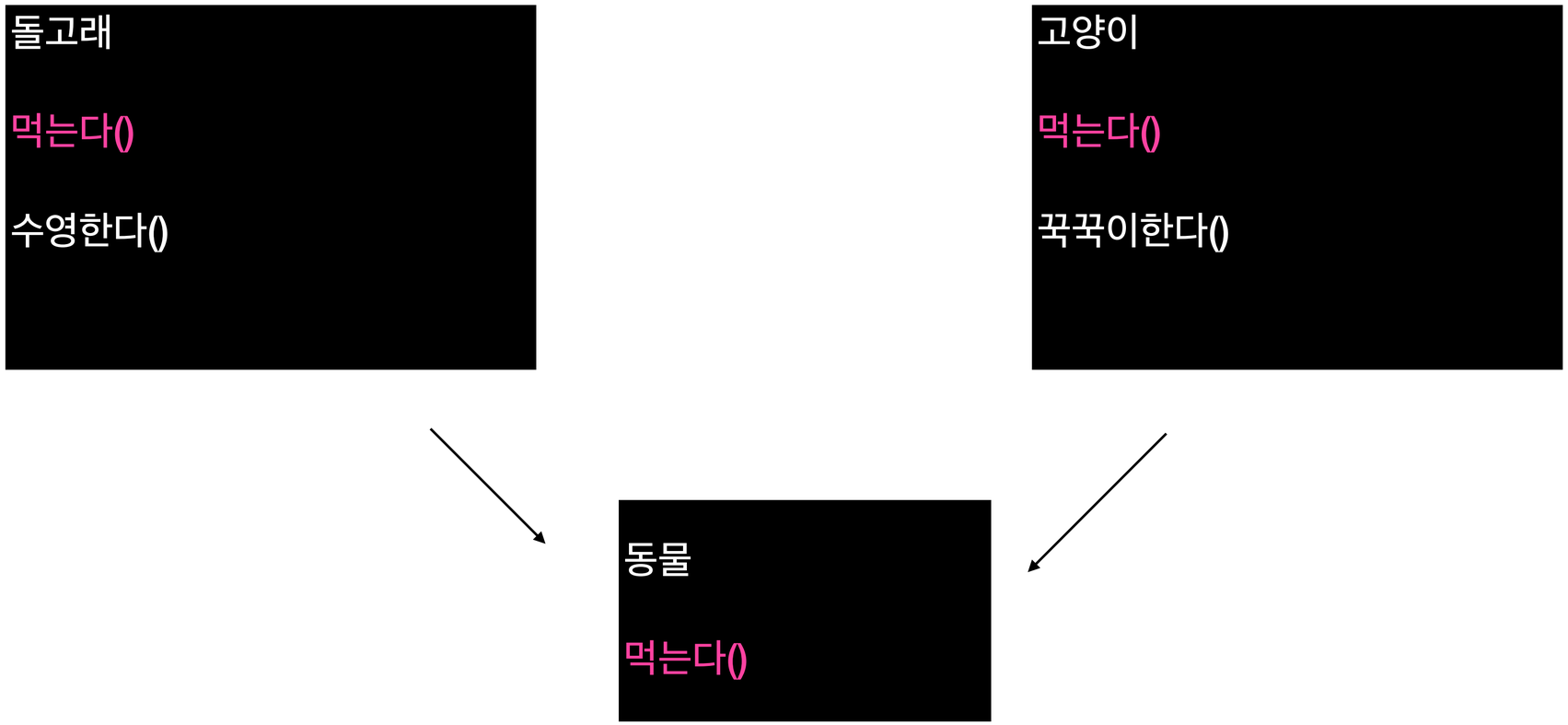
- 클래스
  - 어떤 문제를 해결하기 위한 데이터를 만들기 위해 추상화를 거쳐
    집단에 속하는 속성(attribute)과 행위(behavior)를 변수와 메서드로 정의.
  - 객체를 만들기 위한 메타정보
- 인스턴스 (객체)
  - 클래스에서 정의한 것을 토대로 실제 메모리에 할당한 것
  - 실제 프로그램에서 사용되는 데이터

추상화란?

- 공통의 속성이나 기능을 묶어 이름을 붙이는 것
  - Java의 Interface, 추상 클래스

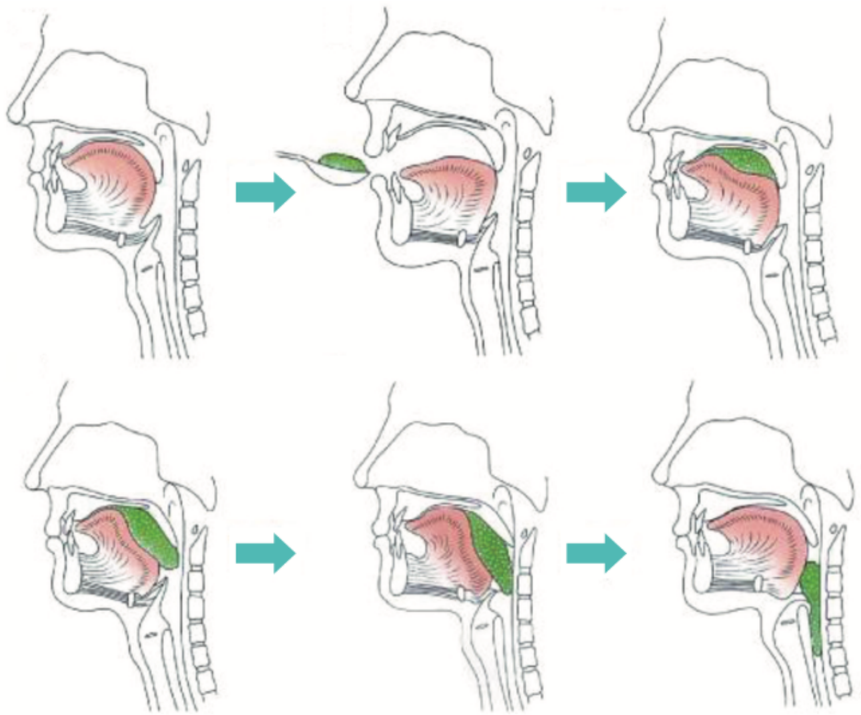
추상화란?

1. 공통된 기능을 뽑아낸다.



추상화란?

2. 현실 세계의 복잡한 현상을 간단한 형태로 모델링(simplify) 한다.



먹는다()

캡슐화란?

- 구현 내용을 내부에 숨겨서 응집도를 높이는 것

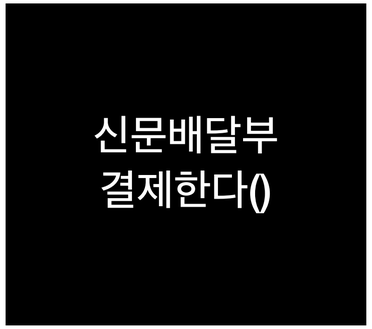
- 응집도
  - 객체가 자신과 관련된 것을 중심적으로 행동한다.
  - 응집도가 높을수록 소프트웨어를 수정할 경우 변경해야될 범위가 명확 -> 좋은 설계
- 결합도
  - 두 객체가 서로 관련되거나 의존하는 정도
  - 결합도가 높을수록 다른 모듈의 변경에 영향을 많이 받게됨 -> 좋은 설계 x

캡슐화란?

- 캡슐화의 목적
  - 코드를 재수정없이 재활용하는 것
  - 접근 제어자를 통한 정보 은닉

- 객체가 외부에 노출하지 않아야할 정보 또는 기능을 접근제어자를 통해 제어 권한이 있는 객체에서만 접근하도록 할 수 있음
- 관련된 기능과 특성을 한곳에 모으고 분류하기 때문에 객체 재활용이 원활

캡슐화를 위반한다면



캡슐화를 위반한다면

신문배달부가

- 고객의 지갑을 가져온다
- 고객의 지갑을 열어서 현금을 확인한다.
- 고객의 지갑에서 돈을 빼간다.

캡슐화를 잘 지킨다면

고객

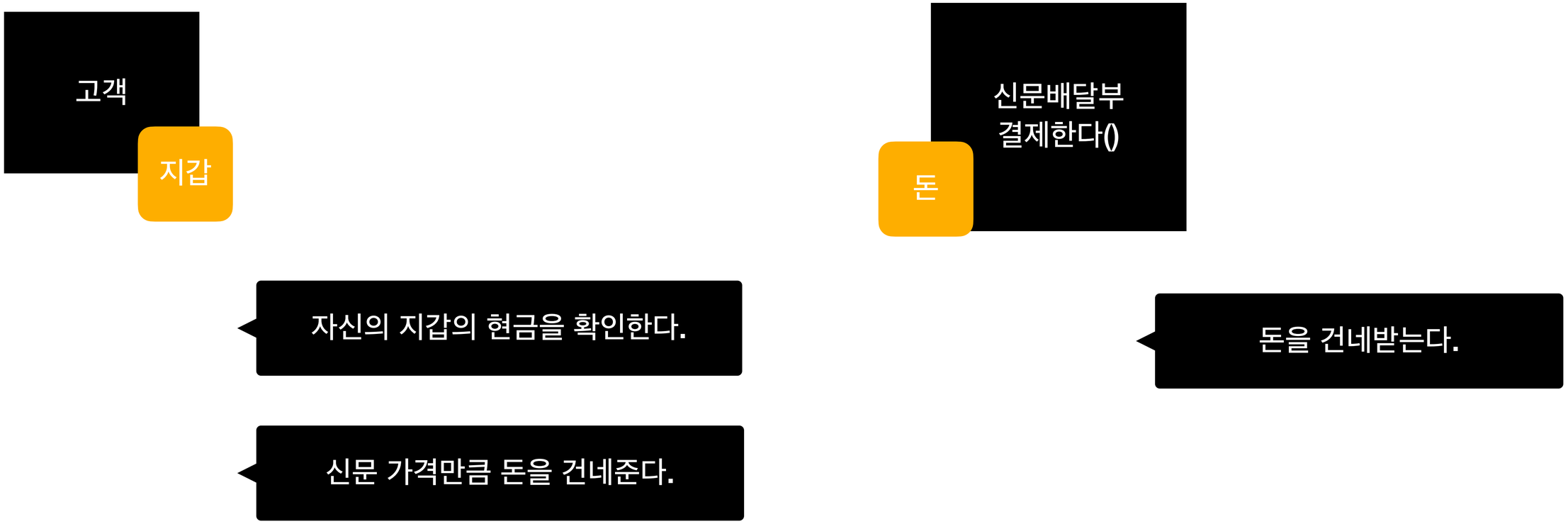
지갑

신문배달부
결제한다()

자신의 지갑의 현금을 확인한다.

신문 가격만큼 돈을 건네준다.

캡슐화를 잘 지킨다면



상속이란?

- 부모클래스의 속성과 기능을 그대로 이어받아 사용할 수 있다.
- 기능의 일부분을 변경해야 할 경우 상속받은 자식클래스에서 해당 기능만 다시 수정하여 사용할 수 있게 하는 것

다형성이란?

- 하나의 변수명, 함수명 등이 상황에 따라 다른 의미로 해석될 수 있는 것
- 서로 다른 유형의 객체가 동일한 메세지에 대해 다르게 반응하기 위해 사용
  - 동일한 메세지를 처리한다 == 같은 역할을 수행한다
  - 다르게 반응한다 == 메세지 처리 방법은 자율적이다.

다형성의 대표적인 예시

- 오버라이딩 : 부모클래스의 메서드와 같은 이름, 매개변수를 재정의하는 것
- 오버로딩 : 같은 이름의 함수를 여러개 정의하고, 매개변수의 타입과 개수를 다르게 하여 매개 변수에 따라 다르게 호출할 수 있게 하는 것