

# Kotlin Class

## class 설계하기

---

### 수업 목표

1. 객체 지향 언어의 의미를 이해한다.
2. 클래스, 객체, 생성자를 이해하고 활용할 수 있다.

## 1. class?

그룹화할 수 있는 변수와 함수를 한군데에 모아 놓고 사용하기 쉽게 이름을 붙여 놓은 것

```
//클래스의 기본 구조
class 클래스명 {
    var 변수
    fun 함수(){
        //코드
    }
}
```

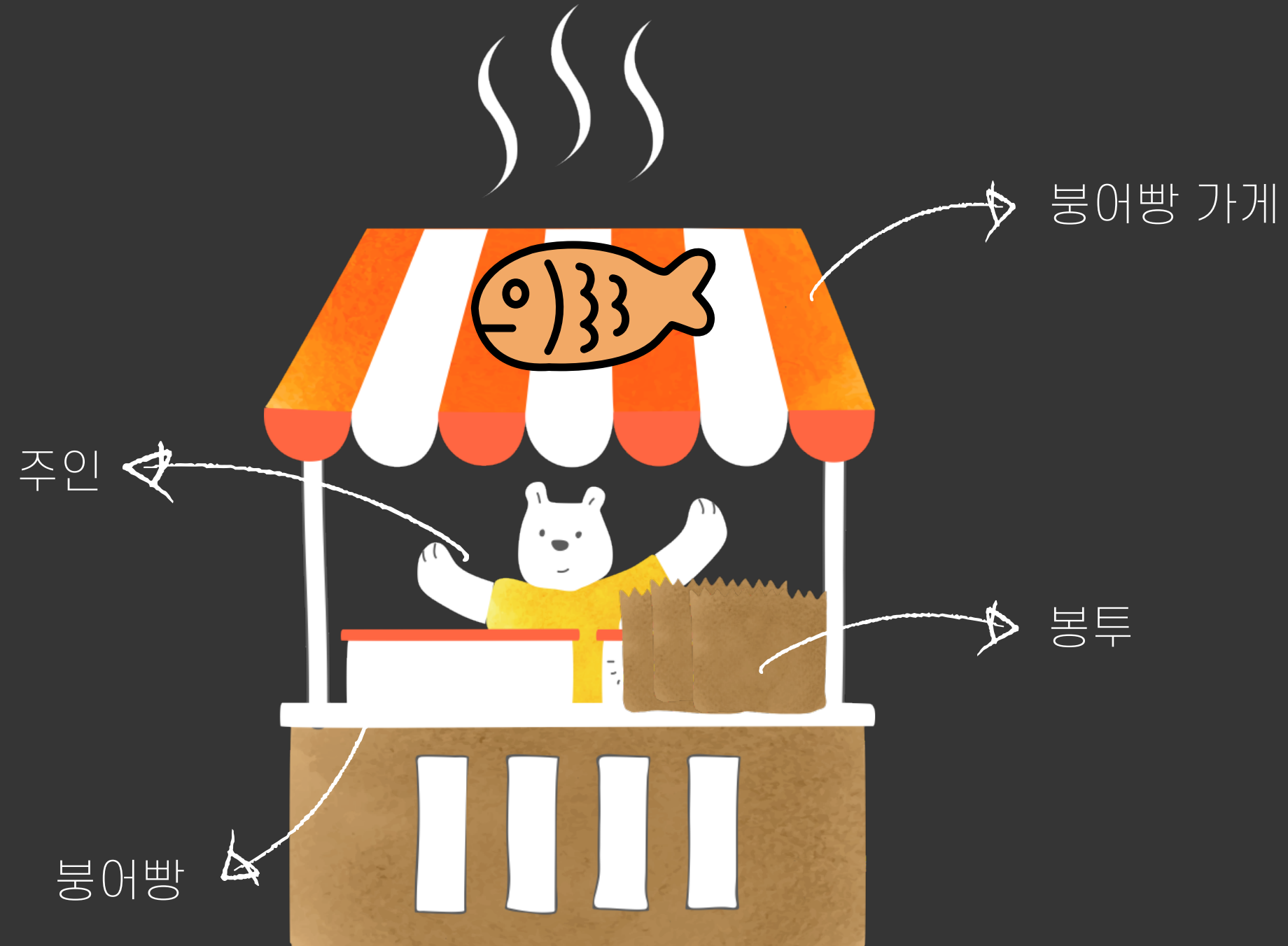
다음은 자동차와 관련된 변수와 함수를 모아둔 class이다.

```
//Car class
class Car {
    val color : String = "black"
    var speed : Int = 0
    fun speedUp(){
        speed += 10
    }
}
```

## 2. 왜 클래스를 사용할까?

우선 객체 지향(OOP)을 간단히 알아보자.

# 객체 지향 (OOP)



현실은 객체들로 이루어져 있다.

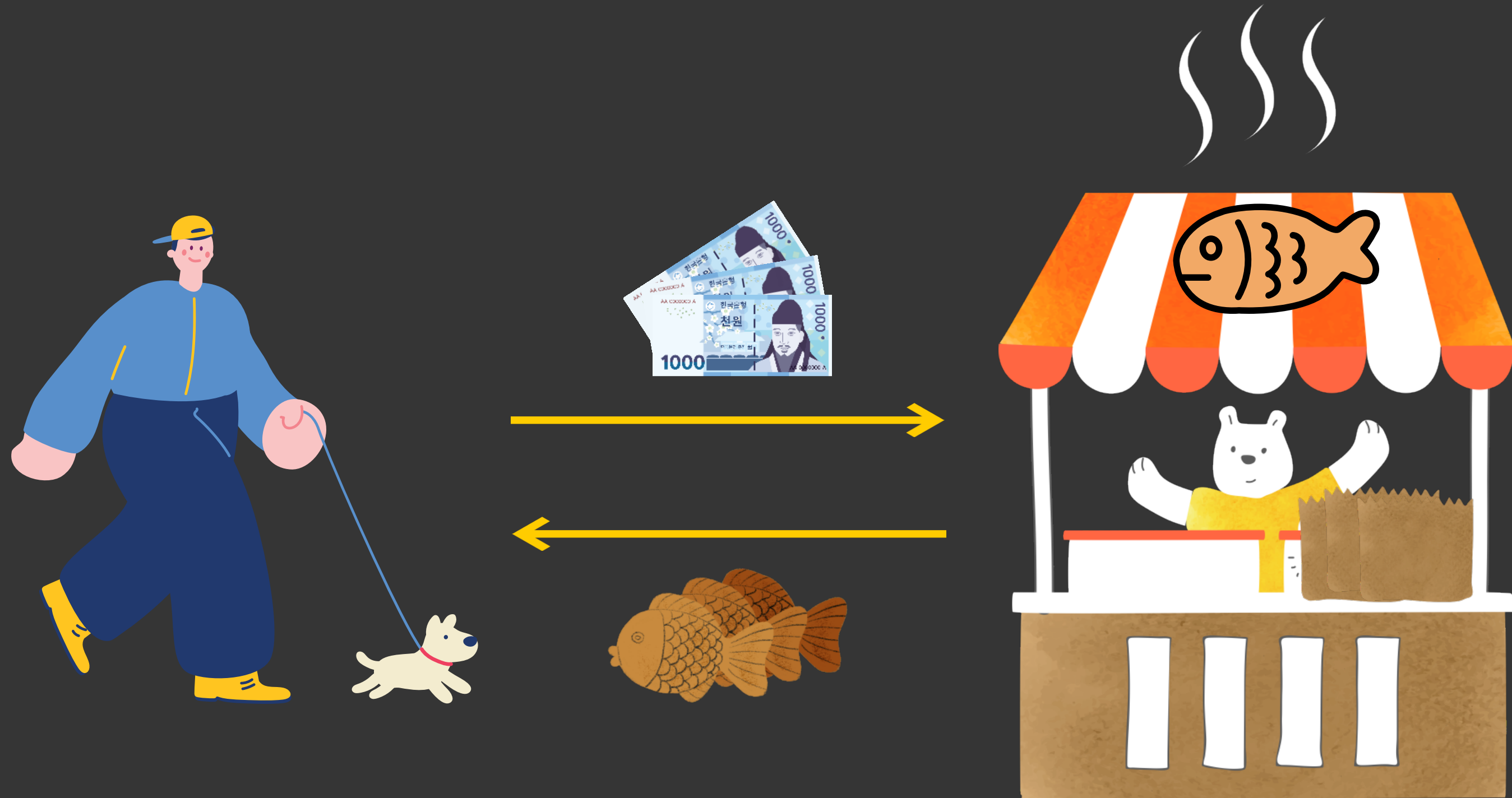
# 객체 지향 (OOP)

객체 : 주인  
속성 : 눈, 코, 입, 손 등등  
행동 : 눈 깜박이기,  
붕어빵 굽기, 돈 세기 등등



객체는 속성(변수)과 행동(함수)으로 이루어져 있다.

# 객체 지향 (OOP)



객체들간에 상호작용이 일어난다.

# 객체 지향 (OOP)

```
val name : String = "owner"  
var age : Int = 20  
var fishCake : Int = 14  
fun sell(val money : Int){  
    //code  
    bag.count--  
}
```



```
val color : String = "brown"  
val size : String = "small"  
var count : Int = 10
```

이러한 특성을 살려서 객체 단위로 프로그래밍하는 것을  
객체 지향 프로그래밍이라고 한다.

## 2. 왜 클래스를 사용할까?

Kotlin은 객체 지향 프로그래밍이 가능한 언어

색상, 브랜드, 주행 기능 등을 가지고 있는 Car class, 눈, 코, 입, 말하기 기능 등을 가지고 있는 Face class 등 객체 단위로 프로그래밍을 할 때 class가 유용하게 사용된다.

클래스는 청사진과 같은 역할을 해, 성질이 같은 객체들을 여러개 만들 때 반복되는 코드를 줄이고 효율성을 높일 수 있다.

ex) 붕어빵 모양 틀(class) -> 붕어빵 객체들을 여러개 만들 수 있다.

코드를 작성해보자.

## 4. class 사용

클래스의 이름에 괄호를 붙여 클래스의 생성자를 호출한다.

```
class FishCake(val flavor : String) {  
    var cost : Int = 500  
    fun printThis(){  
        println("This $flavor fishcake costs $cost")  
    }  
}
```

→ 1. class 생성

```
fun main(){  
    var fishCake = FishCake("민초")  
    fishCake.cost = 1000  
    fishCake.printThis()  
}
```

2. class명 뒤 괄호를 붙이면 그 class의 생성자가 호출이 되며 **instance**가 생성된다.  
생성된 인스턴스는 변수에 담아둘 수 있다. 클래스와 인스턴스의 관계는 붕어빵 틀과 붕어빵과 같은데, class는 붕어빵 틀, instance는 붕어빵이라고 할 수 있다.  
(여기서는 생성자에 flavor 파라미터가 있으므로, "민초" 인자를 넣어주었다)

3. 클래스를 사용한다는 것은 클래스 내부에 정의된 변수와 함수를 사용한다는 의미이다.  
변수에 저장된 인스턴스는 내부에 정의된 변수와 함수를 도트 연산자(.)로 접근할 수 있다.

## data class

간단한 데이터를 저장할 때 사용

data class 클래스명 (val parameter1 : type, var parameter2 : type, var ...)

parameter 앞의 var 또는 val 은 생략할 수 없다

class 앞에 data를 붙이면 data class가 된다

다음은 Player의 이름과 성적을 저장하는 data class 예제이다.

```
data class Player(val name : String, var score : Int)
```

```
fun main(){
```

```
    val firstPlayer = Player("Bob", 12)
```

```
    println(firstPlayer)
```

```
}
```

=> Player(name=Bob, score=12)

이 외에도 추상 클래스(abstract class), 열거 클래스(enum class) 등 코틀린에서 지원하는 특수 클래스가 많으니 관심이 있다면 찾아보자.  
추상 클래스는 다음 시간에 잠깐 다룰 예정!

# 과제

오늘의 과제는 class를 만들고 사용해보는 것입니다.  
과제 조건에 부합하는 코드를 작성해 구글폼에 제출하세요.

## <과제 조건>

1. 생성자가 있는 class를 최소 1개 이상 작성
2. class의 프로퍼티는 최소 2개 이상, 메서드는 1개 이상 작성
3. main 함수에서 인스턴스를 최소 1개 이상의 변수에 담기
4. main 함수에서 클래스의 메서드 사용하기

## <과제 제출 유의사항>

코틀린 파일만 제출하지 말고 전체 폴더를 압축해서 제출

**App:ple Pi**  
made by 이지현

# 수고하셨습니다

과제 제출 폼 : <https://forms.gle/cHBcLUiTWQCCD3dH9>

오늘 수업 / 과제에서 궁금한 점이 있다면 편하게 질문하세요:')

---