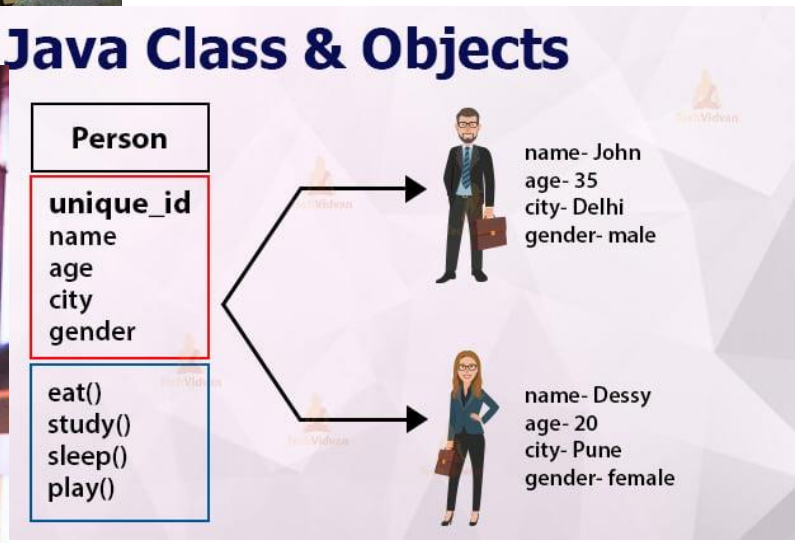
KB

It's Your Life

with

CLASS의 정의


jtbc 금토드라마
이태원 클라쓰
ITAEWON CLASS
2020.1.31 금 PM 10:50 최초공개


Java Class & Objects
Person
unique_id
name
age
city
gender
eat()
study()
sleep()
play()
name- John
age- 35
city- Delhi
gender- male
name- Dessy
age- 20
city- Pune
gender- female


그게
뭔데

Java Class & Objects

Class

Person

Data Members

unique_id
name
age
city
gender

Methods

eat()
study()
sleep()
play()

name- John
age- 35
city- Delhi
gender- male

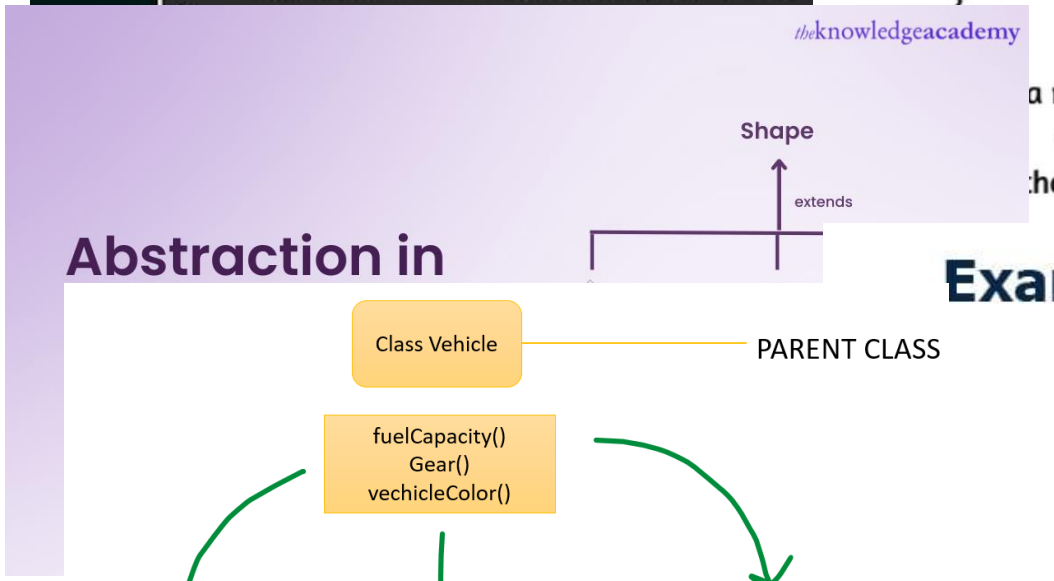
name- Dessy
age- 20
city- Pune
gender- female

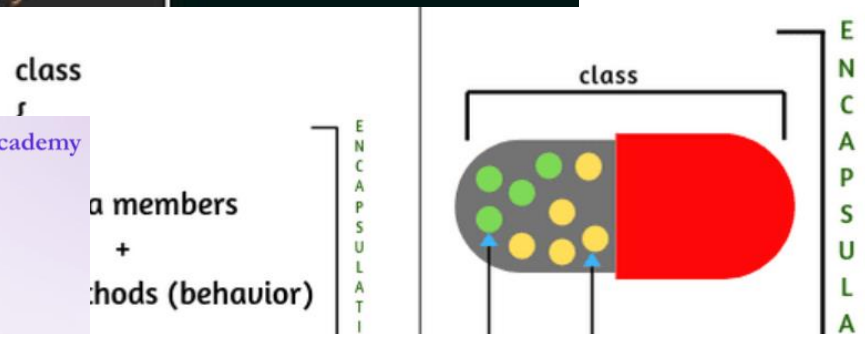
그래서 CLASS 가 뭔가요?

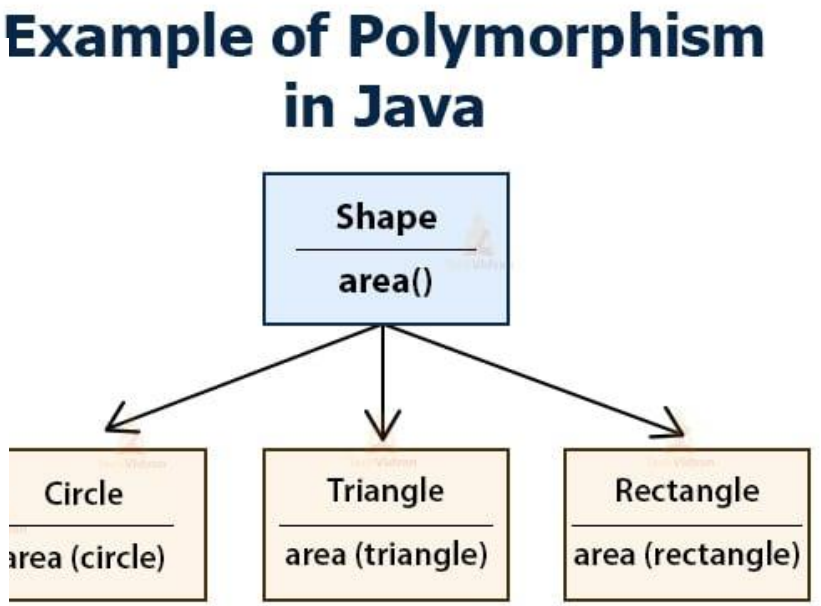
- 이론적으로만 설명을 드리자면……
- CLASS = DATA + METHOD
- 그리고 보통 CLASS 는 설계도이며, 이 설계도를 가지고 만든 실제 제품을 객체(Object) 라고 합니다
- 그리고 이러한 CLASS 를 바탕으로 객체 지향 프로그래밍(OOP)을 구현할 수 있습니다
- OOP 의 4대 중요 개념으로는 캡슐화, 추상화, 상속성, 다형성이 있습니다

Methods in Java

```
public static void main(String[] args) {
    System.out.println("message1");
    sayHi();
    System.out.println("mess
```

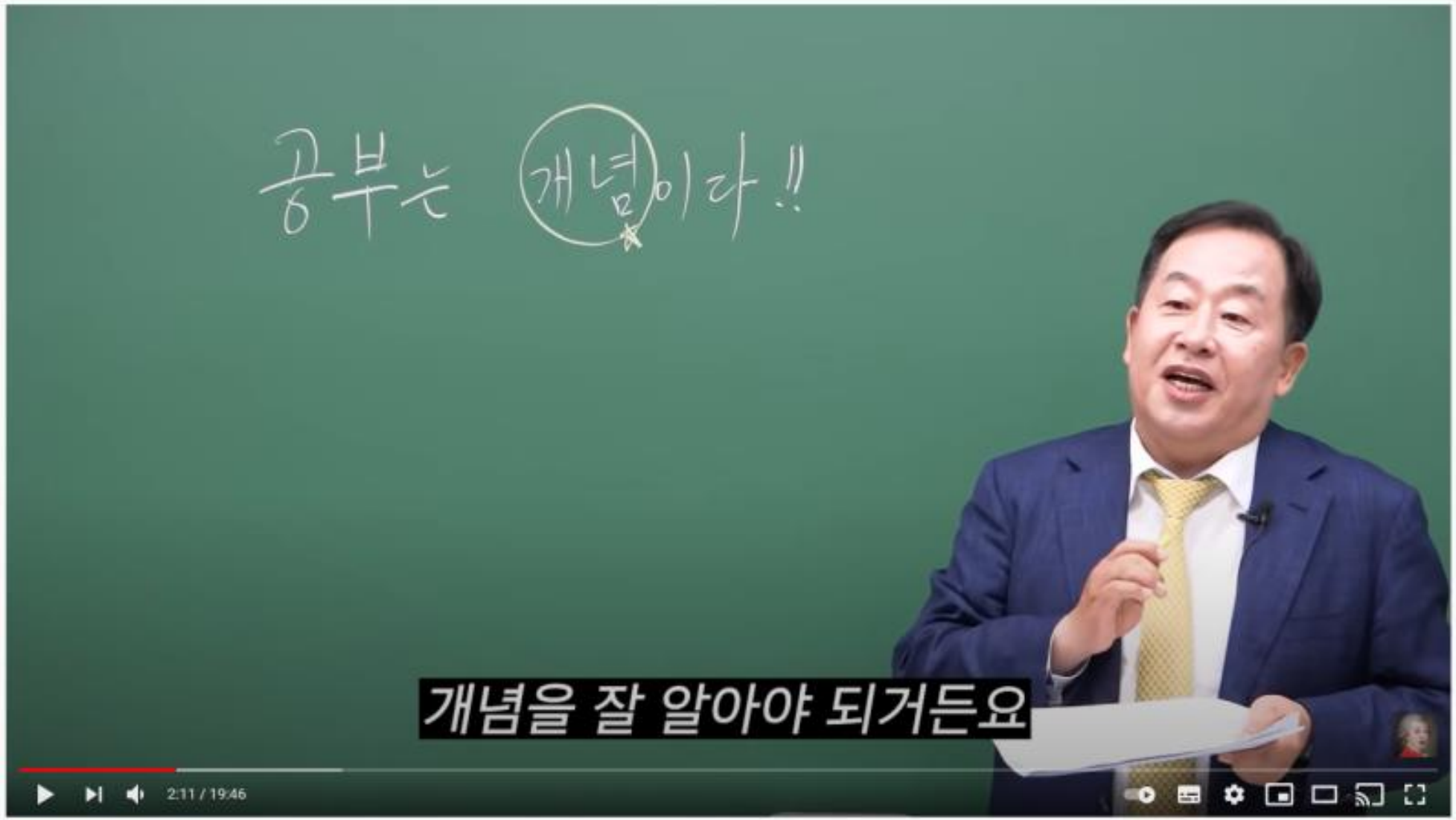






아.....

코딩 어렵네.....



#홍진경 #공부왕찐천재 #손주은

초특급 섭외 손주은 35년 백만불짜리 공부 노하우(쓴소리,명언,치킨이벤트,홍진경) [공부왕찐천재]

조회수 752,966회 • 2021. 8. 21.

1.4만 234 공유 저장 ...


참~ 쉽죠!!

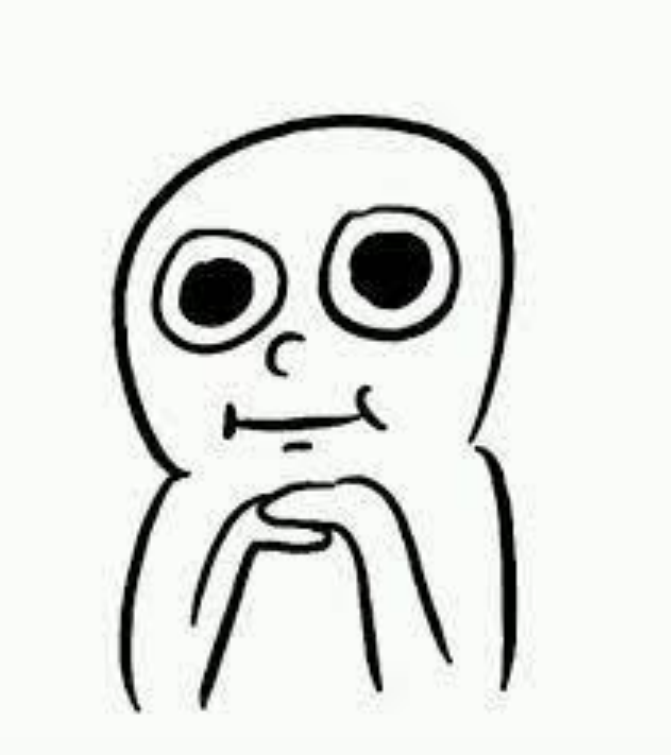
Why?

여러분 기억 나시나요!?

객체(Object)

```
const tetzObj = {
  name: "이효석",
  age: 40,
  isOld: true,

  doLecture: () => {
    console.log("강의!");
  },
};
```



객체를 사용하는 이유!

```
const tetzName = "이효석";
const tetzAge = 40;
const tetzIsOld = true;
const doLectureByTetz = () => {
  console.log("강의!")
}
```

- 갑자기 데이터를 수정해야 한다면?
- 저와 관련 된 모든 변수를 출력해야 한다면?
- 그런데 변수명이 기억나지 않는다면!? 혹시나 하나 빠뜨리게 된다면? 그리고 내가 만든 객체가 아니라면!?

객체를 사용하는 이유!

```
const tetzObj = {
  name: "이효석",
  age: 40,
  isOld: true,
  doLecture: () => {
    console.log("강의!");
  },
};

console.log(tetzObj);
```

- 모든 데이터와 기능(함수)이 하나의 객체에 모여 있어 관리가 쉽습니다!
- 모든 데이터를 출력하기도 쉽고, 까먹을 일도 없습니다!
- 남이 만든 객체라도 아무런 문제 없이 사용할 수 있습니다!

Why?

CLASS

편의점을 운영하게 되었다고 가정해 봅시다

손님을 관리하는 프로그램을 만든다면~?

손님을 관리하는 프로그램을 만든다면~?

```
const customer1 = {
  name: "손흥민",
  age: 31,
  total: 100000,
};

const customer2 = {
  name: "이상혁",
  age: 27,
  total: 300000,
};
```

그런데… 장사가 너무 잘됩니다~!

이제 손님이 1000 명이 넘었습니다...

```
const customer1 = { name: "손흥민", age: 31, total: 100000 };
const customer2 = { name: "이상혁", age: 27, total: 300000 };

// ...
// ...
// ...

const customer1000 = { name: "김수현", age: 36, total: 7000000 };
```

어떤 문제들이 생길까요?

- 새로운 손님이 생길 때마다 개발자가 직접 데이터를 선언해서 만들어 줘야 합니다
- 그러다 가끔 오타가 생기기도 하겠죠? 이런 오타를 찾는 것도 일입니다
- 그리고 손님이 더 늘어서 10만명이 된다면!?

그런데 말입니다……

고객 정보에 Blacklist 를 추가한다면?

```
const customer1 = {                                    , isBlack: false };
const customer2 = {                                    , isBlack: false };

// ...

const customer333 =                                      isBlack: tr_ };
```

하... 이거 누가 대신 해줄 수 없나?

Why!

CLASS

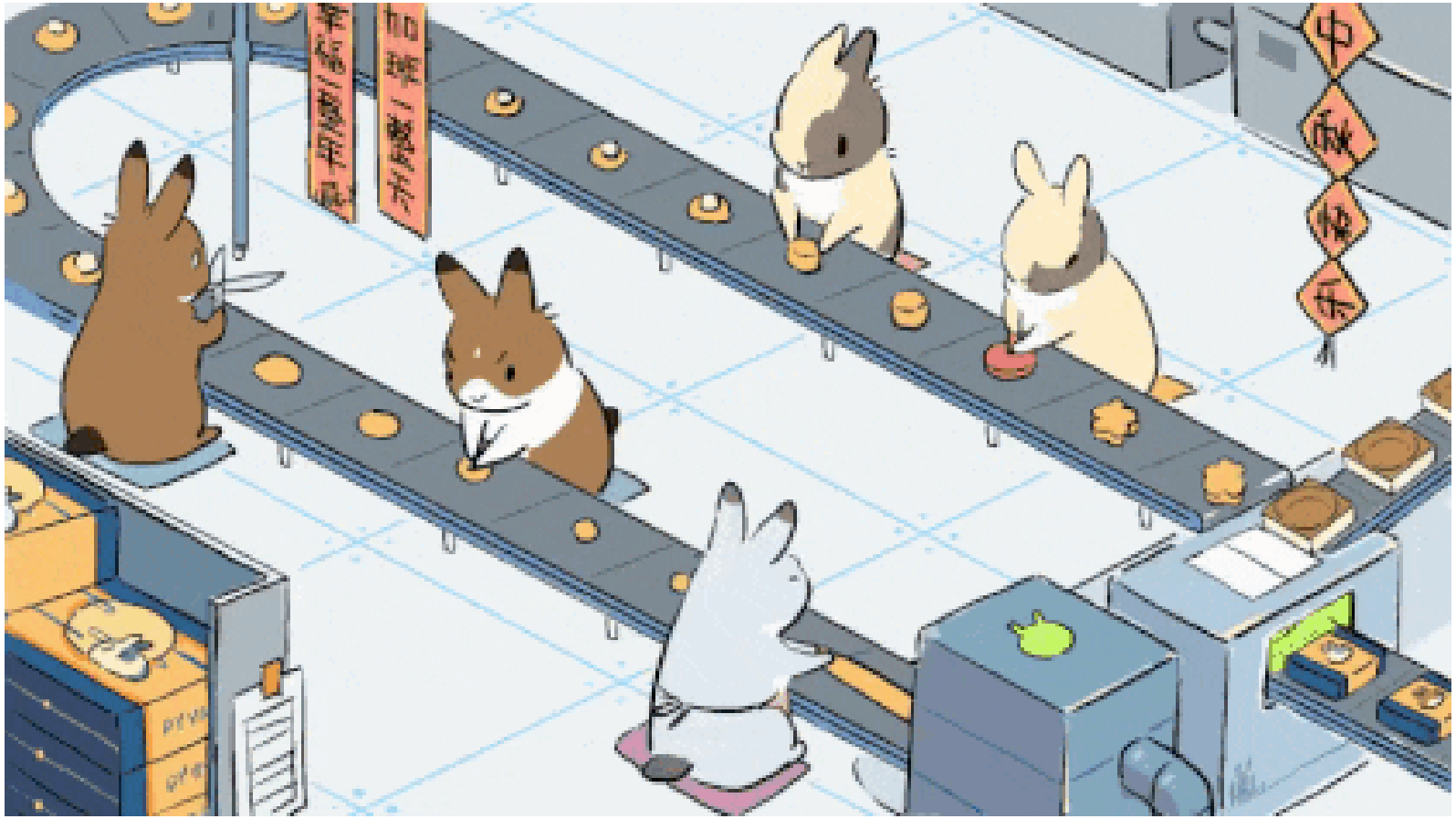
클래스는 객체를 만드는 기계다!

- 설계도를 입력하면 자동으로 객체를 만들어주는 기계를 만들었다 생각해 봅시다!
- 그럼 기계가 저희 대신 객체를 만들어 주겠네요?

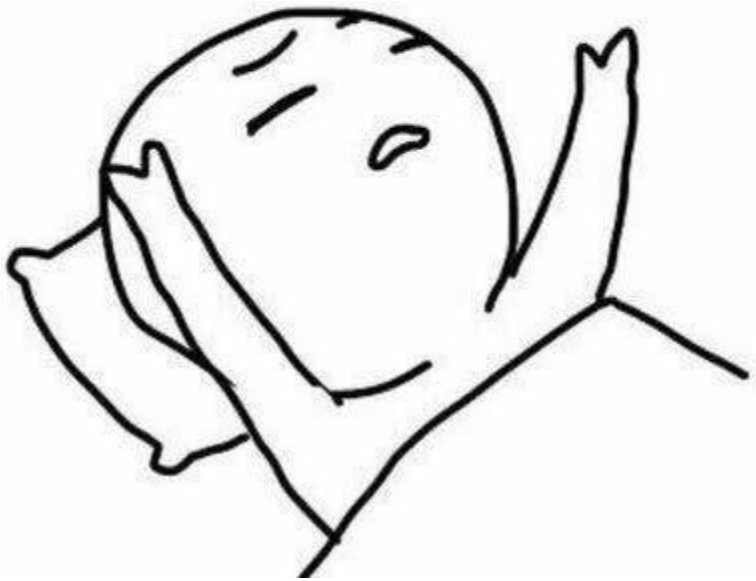
실수!?

오타!?

누락!?



오늘 꿈에 경수가
나오게 해주세요




최초 컨셉안
고객을 만난 후
착공후 도안변경
완공


계획대로야.

WHAT!

CLASS

클래스의 정의

- 클래스는 객체라는 제품을 만들기 위한 설계도 입니다
- 객체(제품) = 속성 + 기능
- 클래스(설계도) = 데이터 + 함수 → 필드 + 생성자 + 메서드
- 클래스의 선언은

```
class 클래스명 {}
```

차두리 설계도

국비

모델명 : Cyberdyne Systems. Model Cha-2(차미네이터)
신 장 : 181cm
무 게 : 대외비(언론에는 적당히 81kg 정도로 공개)
제조사 : *Chaboom Industry*
설계/제작 : 럭키2인자(영플레이 모빌 박사 출신)

기능

속성

iptime 무선공유기 차붐 에디션
(부팅 후 자동으로 차붐의 컨트롤러와 연결됨)
비밀번호 : 00700

모 통신사 유심칩
경기장에선 차붐 유심
집에선 마눌 유심 교체 가능

내장형 안테나

웃는 얼굴
(절전모드가 아니라 화면보호기)

비브라 늄+아다만튬 합금 모노코크 프레임
(결합부위 플리캡 사용 자유로운 관절 움직임 가능)

22번
차드로이드 2.2 버전 업그레이드
센터링 어플 유료 구매

아크 원자로
글리코겐 섭취 발용합을 통해
보약으로 바꿔 체내 저장

치타와 톰슨가젤의 즐기세포 배양 근섬유 이식
(일반인의 5배 / 빅리그 선수의 2배 이상 스피드)

플라이트 파워 유닛+자세 제어용 팬
(가끔 공보다 빨리 뛰어가는 버그를 막기위한 역분사 팬)

클래스의 정의 - 필드

- 필드는 객체에서 속성을 담당하게 되는 값

- **맛 : 팥 | 슈크림**
- **모양 : 붕어 | 잉어**
- **가격 : 500원 | 1000원**
- **만든 날짜 : 2024/6/19**

클래스의 정의 - 필드

- 필드는 클래스 내부에 변수를 선언하듯 사용하면 됩니다~!

```
public class FishBread {  4 usages   new *   1 related problem
    String taste = "팥"; // 붕어빵의 맛 정보를 담당 (팥 / 슈크림 / etc)  1 usage
    String shape = "붕어"; // 붕어빵의 모양 담당 (붕어 / 잉어 / etc)  1 usage
    int price = 500; // 붕어빵의 가격 담당  1 usage
    Date makeTime = new Date(); // 붕어빵이 만들어진 시간 설정  1 usage
}
```

붕어빵의 속성(= 필드)을 설정

HOW

CLASS


MBC

CLASS (=인스턴스화)

DATA

OBJECT

그럼 붕어빵을 찍어 봅시다~!

- 클래스로 객체를 만들면 객체의 데이터 타입 자체는 클래스가 됩니다
- 새로운 클래스를 선언 할 때는 new 라는 예약어를 사용 합니다

위의 과정을 어렵게 풀어서 쓰면~!

- 여러분은 3가지 용어, 객체 / 클래스 / 인스턴스의 정의를 배워야 합니다
- 객체 : 모든 인스턴스를 대표하는 '일반적' 용어
- 클래스 : 객체를 만들기 위한 설계도
- 인스턴스 : 특정 클래스로 생성된 객체

인스턴스화

클래스 ⟶ **인스턴스(객체)**

고양이 사용법
(미치셨습니까, 휴먼)

FishBread 클래스 사용을 위한 운영 클래스 만들기

FishBread 클래스를 가져와서 FishBread 타입의 객체를 생성 후 fish1 이라는 변수에 주소 저장!


kb-java-lecture C:₩git₩kb-java-le
.idea
out
src
array
classs
FishBread
FishBreadMain
scanner
variable
4
5 public class FishBreadMain { new *
6 public static void main(String[] args) { new *
7 FishBread fish1 = new FishBread();
8
9 fish1.taste = "팥";
10 fish1.shape = "붕어";
11 fish1.price = 200;
12 }
13 }
FishBread 의 속성(=필드)의 값을 설정!

```java
public class FishBreadMain {  new *
    public static void main(String[] args) {  new *
        FishBread fish1 = new FishBread();

        fish1.taste = "팥";
        fish1.shape = "붕어";
        fish1.price = 200;

        System.out.println("붕어빵의 맛은? : " + fish1.taste);
        System.out.println("붕어빵의 모양은? : " + fish1.shape);
        System.out.println("붕어빵의 가격은? : " + fish1.price);
        System.out.println("붕어빵의 생산 시각은? : " + fish1.makeTime);
    }
}
```

설정한 속성 값에 접근하여 값을 출력!

```
붕어빵의 맛은? : 팥
붕어빵의 모양은? : 붕어
붕어빵의 가격은? : 200
붕어빵의 생산 시각은? : Wed Jun 19 11:57:45 KST 2024
```

메모리의 상태

```java
public class FishBreadMain {  new *
    public static void main(String[] args) {  new *
        FishBread fish1 = new FishBread();

        fish1.taste = "팥";
        fish1.shape = "붕어";
        fish1.price = 200;

    }
}
```

```
5D E9 7A FF  FF FF 66 2E  0F 1F 84 00  00 00 00 00  55 48 89 E5  ].z...f.........UH..
48 83 EC 30  F2 0F 11 45  E0 F2 0F 11  4D E8 F2 0F  11 55 D0 F2  H..0...E....M....U..
0F 11 5D D8  F2 0F 10 45  E0 F2 0F 10  4D E8 F2 0F  10 55 D0 F2  ..]....E....M....U..
0F 10 5D D8  F2 0F 58 C2  F2 0F 58 CB  48 8D 7D F0  E8 3F 7F 48  ..]...X...X.H.}..?.H
00 F2 0F 10  45 F0 F2 0F  10 4D F8 48  83 C4 30 5D  C3 66 66 66  ....E....M.H..0].fff
66 66 66 2E  0F 1F 84 00  00 00 00 00  55 48 89 E5  48 83 EC 40  fff.........UH..H..@
F2 0F 11 45  E0 F2 0F 11  4D E8 F2 0F  11 55 D0 F2  0F 11 5D D8  ...E....M....U....].
F2 0F 10 45  E0 F2 0F 10  4D E8 F2 0F  10 55 D0 F2  0F 10 5D D8  ...E....M....U....].
0F 28 E0 F2  0F 59 E2 0F  28 E9 F2 0F  59 EB F2 0F  5C E5 F2 0F  .(...Y..(...Y...\...
59 C3 F2 0F  59 CA F2 0F  58 C1 48 8D  7D F0 F2 0F  11 45 C8 0F  Y...Y...X.H.}....E..
28 C4 F2 0F  10 4D C8 E8  BC 7E 48 00  F2 0F 10 45  F0 F2 0F 10  (....M...~H....E....
4D F8 48 83  C4 40 5D C3  66 66 66 2E  0F 1F 84 00  00 00 00 00  M.H..@].fff.........
55 48 89 E5  48 81 EC 80  00 00 00 F2  0F 11 45 E0  F2 0F 11 4D  UH..H.........E....M
```

```java
public class FishBreadMain {  new *
    public static void main(String[] args) {  new *
        FishBread fish1 = new FishBread();

        fish1.taste = "팥";
        fish1.shape = "붕어";
        fish1.price = 200;

    }
}
```

fish1(x001)

```
FF FF 66 2E  0F 1F 84 00  00 00 00 00  55 48 89 E5  ].z...f.........UH..
F2 0F 11 45  E0 F2 0F 11  4D E8 F2 0F  11 55 D0 F2  H..0...E....M....U..
0F 11 5D D8  F2 0F 10 45  E0 F2 0F 10  4D E8 F2 0F  10 55 D0 F2  ..]....E....M....U..
F2 0F 58 C2  F2 0F 58 CB  48 8D 7D F0  E8 3F 7F 48  ..]...X...X.H.}..?.H
00 F2 0F 10  45 F0 F2 0F  10 4D F8 48  83 C4 30 5D  C3 66 66 66  ....E....M.H..0].fff
66 66 66 2E  0F 1F 84 00  00 00 00 00  55 48 89 E5  48 83 EC 40  fff.........UH..H..@
F2 0F 11 45  E0 F2 0F 11  4D E8 F2 0F  11 55 D0 F2  0F 11 5D D8  ...E....M....U....].
F2 0F 10 45  E0 F2 0F 10  4D E8 F2 0F  10 55 D0 F2  0F 10 5D D8  ...E....M....U....].
0F 28 E0 F2  0F 59 E2 0F  28 E9 F2 0F  59 EB F2 0F  5C E5 F2 0F  .(...Y..(...Y...\...
59 C3 F2 0F  59 CA F2 0F  58 C1 48 8D  7D F0 F2 0F  11 45 C8 0F  Y...Y...X.H.}....E..
28 C4 F2 0F  10 4D C8 E8  BC 7E 48 00  F2 0F 10 45  F0 F2 0F 10  (....M...~H....E....
4D F8 48 83  C4 40 5D C3  66 66 66 2E  0F 1F 84 00  00 00 00 00  M.H..@].fff.........
55 48 89 E5  48 81 EC 80  00 00 00 F2  0F 11 45 E0  F2 0F 11 4D  UH..H.........E....M
```

```java
public class FishBreadMain {  new *
    public static void main(String[] args) {  new *
        FishBread fish1 = new FishBread();

        fish1.taste = "팥";
        fish1.shape = "붕어";
        fish1.price = 200;
```

fish1(x001)

```
FF FF 66 2E  0F 1F 84 00  00 00 00 00  55 48 89 E5  ].z...f.........UH..
F2 0F 11 45  E0 F2 0F 11  4D E8 F2 0F  11 55 D0 F2  H..0...E....M....U..
0F 11 5D D8  F2 0F 10 45  E0 F2 0F 10  4D E8 F2 0F  10 55 D0 F2  ..]....E....M....U..
F2 0F 58 C2  F2 0F 58 CB  48 8B 7D F0  48 89 45 E8  ..]...X...X.H.}..?.H
00 F2 0F 10  45 F0 F2 0F  10 4D F8 48  8B 45 E8 48  ....E....M.H..0].fff
66 66 66 2E  0F 1F 84 00  00 00 00 00  55 48 89 E5  fff.........UH..H..@
F2 0F 11 45  E0 F2 0F 11  4D E8 F2 0F  11 55 D0 F2  ...E....M....U....].
F2 0F 10 45  E0 F2 0F 10  4D E8 F2 0F  10 55 D0 F2  ...E....M....U....].
0F 28 E0 F2  0F 59 E2 0F  28 E9 F2 0F  59 EB F2 0F  5C E5 F2 0F  .(...Y..(...Y...\...
59 C3 F2 0F  59 CA F2 0F  58 C1 48 8D  7D F0 F2 0F  11 45 C8 0F  Y...Y...X.H.}....E..
28 C4 F2 0F  10 4D C8 E8  BC 7E 48 00  F2 0F 10 45  F0 F2 0F 10  (....M...~H....E....
4D F8 48 83  C4 40 5D C3  66 66 66 2E  0F 1F 84 00  00 00 00 00  M.H..@].fff.........
55 48 89 E5  48 81 EC 80  00 00 00 F2  0F 11 45 E0  F2 0F 11 4D  UH..H.........E....M
```

taste = 팥
shape = 붕어
price = 200

소개팅은 주선자의
백마디 말보다
한번 만나봐야
견적이 나온다.

실습, 편의점 고객 클래스 만들어 보기!

- 그럼 이번에는 편의점 고객 정보 객체를 만들어주는 클래스를 한번 만들어 봅시다!
- 클래스명 : Customer
- 고객 필드
  - 고객 이름(name)
  - 나이(age)
  - 총 사용한 금액(total)
  - blacklist 여부(blacklist)

실습, 편의점 고객 클래스 만들어 보기!

- 고객 클래스를 사용할 수 있는 CustomerMain 클래스 만들기
- 고객 클래스를 사용해서 고객 1명 만들어서 customer1 변수에 담기 (= 고객 1명을 인스턴스화 하여 customer1 변수에 담기)
- 해당 고객 객체의 정보를 자신의 정보로 입력하기
- 해당 고객 객체의 정보를 출력하는 프로그램 작성하기

```
고객의 이름은? : 이효석
고객의 나이는? : 40
고객이 총 사용한 금액은? : 10000000
고객은 블랙리스트인가요!? : true
```

잠깐...!

아직 부족한 점이 많아 보입니다~!

- 일단 이전에 문제와 동일하게 현재 모든 객체의 속성을 바꾸려면 하나하나 접근해서 변경해 줘야 합니다
- 그리고 클래스는 객체를 찍어내는 기계라고 배웠는데... 이게 기계가 맞나 싶습니다

```java
public class FishBreadMain {  new *
    public static void main(String[] args) {  new *
        FishBread fish1 = new FishBread();

        fish1.taste = "팥";
        fish1.shape = "붕어";
        fish1.price = 200;

        System.out.println("붕어빵의
        System.out.println("붕어빵의
        System.out.println("붕어빵의
        System.out.println("붕어빵의
    }
}
```

생성자를

배울 시간!

인스턴스를 만들 때, 속성을 지정해 봅시다

- 지금까지는 클래스가 기계처럼 작동하지는 않았습니다.
- 지금 우리에게 필요한 것은? 인스턴스를 생성할 때, 해당 인스턴스의 속성을 지정하고 싶은 것입니다~!
- 따라서 우리는 인스턴스를 처음 만들 때, 초기값을 저장하는 역할을 하는 생성자를 배워서 해당 문제를 해결해 볼 겁니다~!

클래스에 생성자 선언해보기

```
public class FishBread {  4 usages   new *   1 related problem
    String taste = "팥"; // 붕어빵의 맛 정보를 담당 (팥 / 슈크림 / 
    String shape = "붕어"; // 붕어빵의 모양 담당 (붕어 / 잉어 / etc
    int price = 500; // 붕어빵의 가격 담당  2 usages
    Date makeTime = new Date(); // 붕어빵이 만들어진 시간 설정  1 us

    FishBread(String taste, String shape, int price) {  2 usages
        this.taste = taste;
        this.shape = shape;
        this.price = price;
    }
```

클래스와 동일한 이름 사용

자기 자신을 가르키는
this 를 사용하여

각각의 필드에 접근하여
매개변수로 받은 값을
초기 값으로 지정

생성자 사용해보기

```
public class FishBreadMain {  new *
    public static void main(String[] args) {  new *
        FishBread fish1 = new FishBread( taste: "슈크림",  shape: "잉어",  price: 500);


//          fish1.taste = "팥";
//          fish1.shape = "붕어";
//          fish1.price = 200;
```

생성자에게 각각의 매개변수를 전달하여
새롭게 생성되는 인스턴스의 초기값을 지정

전 처럼 직접 인스턴스의
필드에 접근하여 값 지정 X

초기값을 넣어야만 하나요!?

- 이렇게 생성자를 사용하면 모든 클래스를 만들 때마다 매개변수를 입력해 줘야 합니다
- 그런데 현재 우리 붕어빵집에서 제일 잘 팔리는 붕어빵은 500원 짜리 팥 붕어빵입니다
- 이전 처럼 초기값을 설정해둔 그대로 사용하면 안되나요!?
- 그럼, 이전 초기값을 사용하는 fish2 인스턴스를 만들어 봅시다

```java
public class FishBreadMain {  new *
    public static void main(String[] args) {  new *
        FishBread fish1 = new FishBread( taste: "슈크림",  shape: "잉어",  price: 500);

        FishBread fish2 = new FishBread();
```

```
java: constructor Fishbread in class Fishbread cannot be applied to given types;
  required: java.lang.String,java.lang.String,int,java.util.Date
  found:    no arguments
  reason: actual and formal argument lists differ in length
```


OTL

그 원인은 클래스 정의에 있습니다!

```
public class FishBread {  4 usages   new *   1 related problem
    String taste = "팥"; // 붕어빵의 맛 정보를 담당 (팥 / 슈크림 / e
    String shape = "붕어"; // 붕어빵의 모양 담당 (붕어 / 잉어 / etc
    int price = 500; // 붕어빵의 가격 담당  2 usages
    Date makeTime = new Date(); // 붕어빵이 만들어진 시간 설정  1 us

    FishBread(String taste, String shape, int price) {  2 usages
        this.taste = taste;
        this.shape = shape;
        this.price = price;
    }
```

지금은 매개변수가 없는
생성자가 존재하지 않는 상황!

따라서 에러가 발생 합니다!

그럼 만들어 봅시다~!

```
public class FishBread {  4 usages   new *
    String taste = "팥"; // 붕어빵의 맛 정보를 담당 (팥 / 슈크
    String shape = "붕어"; // 붕어빵의 모양 담당 (붕어 / 잉어
    int price = 500; // 붕어빵의 가격 담당  2 usages
    Date makeTime = new Date(); // 붕어빵이 만들어진 시간 설

    FishBread() {}  usage   new *

    FishBread(String taste, String shape, int price) {
        this.taste = taste;
        this.shape = shape;
        this.price = price;
    }
}
```

아무런 매개변수를 받지 않는
생성자를 선언해 봅시다!!

```java
public class FishBreadMain {  new *
    public static void main(String[] args) {  new *
        FishBread fish1 = new FishBread( taste: "슈크림",  shape: "잉어",  price: 500);
        System.out.println("붕어빵의 맛은? : " + fish1.taste);
        System.out.println("붕어빵의 모양은? : " + fish1.shape);
        System.out.println("붕어빵의 가격은? : " + fish1.price);
        System.out.println("붕어빵의 생산 시각은? : " + fish1.makeTime);

        FishBread fish2 = new FishBread();
        System.out.println("붕어빵의 맛은? : " + fish2.taste);
        System.out.println("붕어빵의 모양은? : " + fish2.shape);
        System.out.println("붕어빵의 가격은? : " + fish2.price);
        System.out.println("붕어빵의 생산 시각은? : " + fish2.makeTime);
    }
}
```

빈 생성자가 있기 때문에
기본 설정 값으로 생성

```
붕어빵의 맛은? : 슈크림
붕어빵의 모양은? : 잉어
붕어빵의 가격은? : 500
붕어빵의 생산 시각은? : Wed Jun 19 12:5
붕어빵의 맛은? : 팥
붕어빵의 모양은? : 붕어
붕어빵의 가격은? : 500
붕어빵의 생산 시각은? : Wed Jun 19 12:5
```

위의 개념을 어렵게 풀어서 쓰면~!

- 위와 같은 개념을 어렵게 풀어서 쓰면 ‘오버로딩‘ 이라고 합니다~!
- 오버로딩 : 같은 이름이라도 매개변수의 유형 또는 개수가 다른 경우 다른 방법을 적용

잠깐...!

```java
public class FishBreadMain {  new *
    public static void main(String[] args) {  new *
        FishBread fish1 = new FishBread( taste: "슈크림",  shape: "잉어",  price: 500);
        System.out.println("붕어빵의 맛은? : " + fish1.taste);
        System.out.println("붕어빵의 모양은? : " + fish1.shape);
        System.out.println("붕어빵의 가격은? : " + fish1.price);
        System.out.println("붕어빵의 생산 시각은? : " + fish1.makeTime);

        FishBread fish2 = new FishBread();
        System.out.println("붕어빵의 맛은? : " + fish2.taste);
        System.out.println("붕어빵의 모양은? : " + fish2.shape);
        System.out.println("붕어빵의 가격은? : " + fish2.price);
        System.out.println("붕어빵의 생산 시각은? : " + fish2.makeTime);
    }
}
```

매번 이렇게 반복해서
써줘야만 할까요!?

매서드를

배울 시간!

반복적으로 사용되는 기능을 처리해 봅시다!

- 이렇게 반복되는 기능이 있을 경우, 하나의 함수로 만들어 사용하면 편하다는 것은 이미 배우셨을 겁니다!
- 클래스에서는? 필드와 함께 필드를 사용하는 함수(메서드) 까지 클래스에 넣어 버리면 됩니다!

클래스에 메서드 선언해보기

```
FishBread(String taste, String shape, int price) {  1 usage   new *
    this.taste = taste;
    this.shape = shape;
    this.price = price;
}

void printFishBread() {  no usages   new *
    System.out.println("붕어빵의 맛은? : " + this.taste);
    System.out.println("붕어빵의 모양은? : " + this.shape);
    System.out.println("붕어빵의 가격은? : " + this.price);
    System.out.println("붕어빵의 생산 시각은? : " + this.makeTime);
}
```

this 를 사용해서
어떤 인스턴스에서 호출이 되건
해당 인스턴스 스스로의
필드 값을 출력하도록 만들기!

클래스 메서드 사용해보기!

```
public class FishBreadMain {  new *
    public static void main(String[] args) {  new *
        FishBread fish1 = new FishBread( taste: "슈크림",  sha
        fish1.printFishBread();

        FishBread fish2 = new FishBread();
        fish2.printFishBread();
    }
}
```

클래스에서 정의한 메서드를 사용

→ 인스턴스의 필드를 출력

```
붕어빵의 맛은? : 슈크림
붕어빵의 모양은? : 잉어
붕어빵의 가격은? : 500
붕어빵의 생산 시각은? : Wed Jun 19 1
붕어빵의 맛은? : 팥
붕어빵의 모양은? : 붕어
붕어빵의 가격은? : 500
붕어빵의 생산 시각은? : Wed Jun 19 1
```

실습, 편의점 클래스 작성하기!

- 방금 실습에서 작성 했던 편의점 클래스를 완성해 봅시다
- 편의점의 새로운 고객 인스턴스를 만들 때, 이름 / 나이 / 총 사용 금액 / 블랙 리스트 여부를 전달하는 생성자를 만들어 봅시다!
- 편의점 고객 인스턴스 생성 시, 아무런 매개변수 전달이 없을 경우 이름은 “아직 몰라요” / 나이는 0 / 총 사용 금액은 0 / 블랙 리스트 여부는 false 로 만드는 기능을 추가해 봅시다!
- 편의점 특정 고객 인스턴스의 정보를 출력하는 메서드를 추가해 봅시다

실습, 편의점 클래스 작성하기!

- 아래의 운영 클래스 코드를 실행 하였을 때, 다음과 같은 결과가 출력!

```
public class CustomerMain {  new *
    public static void main(String[] args) {  new *
        Customer customer1 = new Customer( name: "이효석",  age: 40,  total: 10000,  isBlackList: true);
        customer1.printCustomerInfo();


        Customer customer2 = new Customer();
        customer2.printCustomerInfo();
    }
}
```

```
고객의 성함은 : 이효석
고객의 나이는 : 40
고객의 총 사용 금액은 : 10000
고객의 블랙리스트 여부는 : true
고객의 성함은 : 아직 몰라요
고객의 나이는 : 0
고객의 총 사용 금액은 : 0
고객의 블랙리스트 여부는 : false
```


수고하셨습니다!