Introduction to
OOP with Java

Sheikh Azizul Hakim

1705002@ugrad.cse.buet.ac.bd

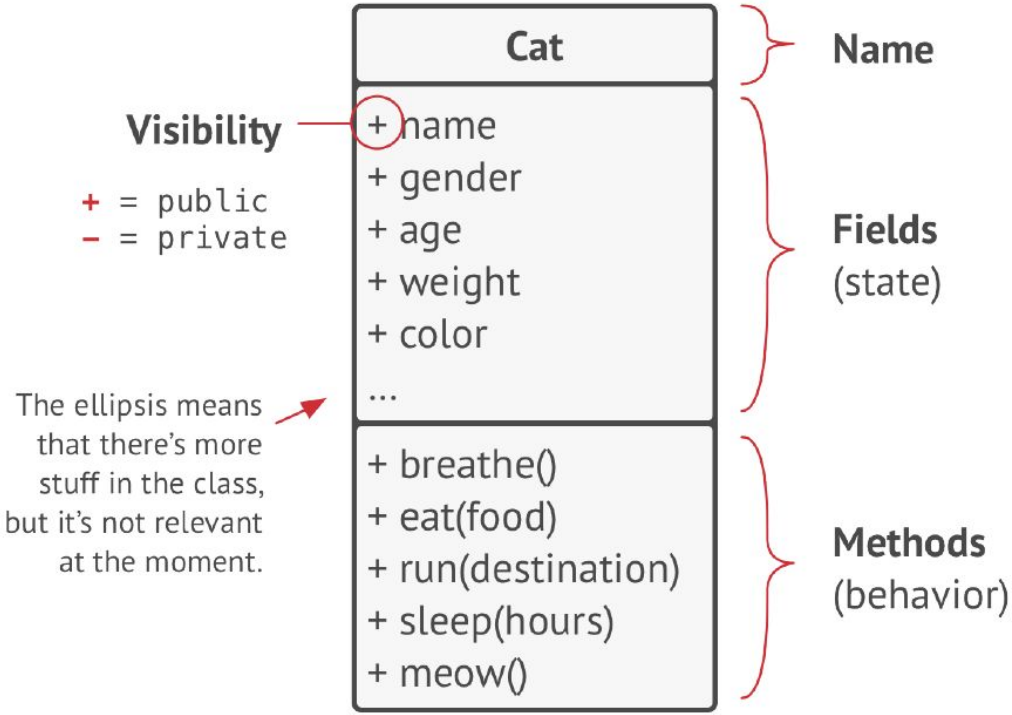
Tentative Contents

1. Some basic Java syntax
2. Language agnostic OOP
3. Java Classes
4. Java inheritance
5. Exceptions in Java
6. Generics and Collections Framework
7. File I/O

Objects and Classes

A class is a collection of some variables and methods behind an abstract idea.

An object is an “instance” of that idea.



Example of a Class

Oscar: Cat

```
name    = "Oscar"
sex     = "male"
age     =  3
weight  =  7
color   =  brown
texture =  striped
```

Objects are instances of classes.

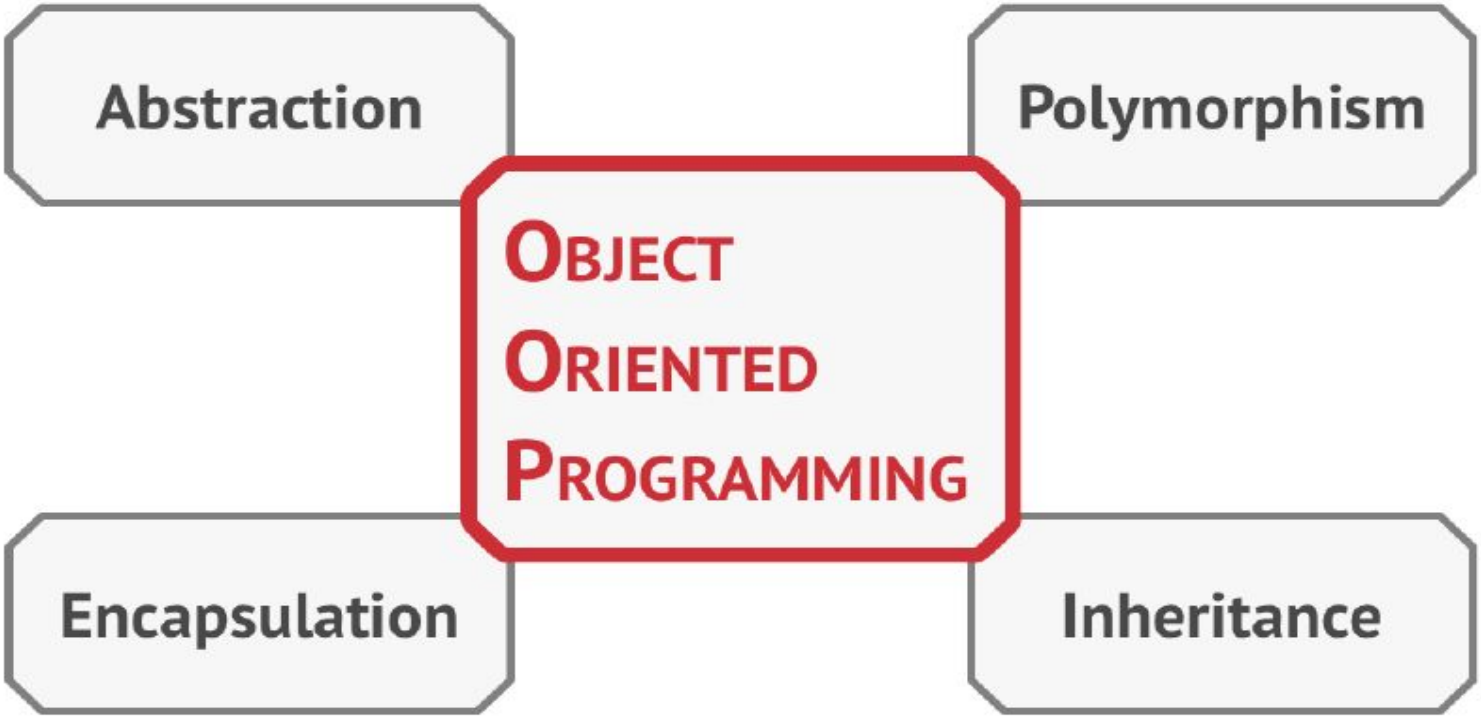
Oscar: Cat

```
name    = "Oscar"
sex     = "male"
age     =  3
weight  =  7
color   =  brown
texture =  striped
```

Luna: Cat

```
name    = "Luna"
sex     = "female"
age     =  2
weight  =  5
color   =  gray
texture =  plain
```

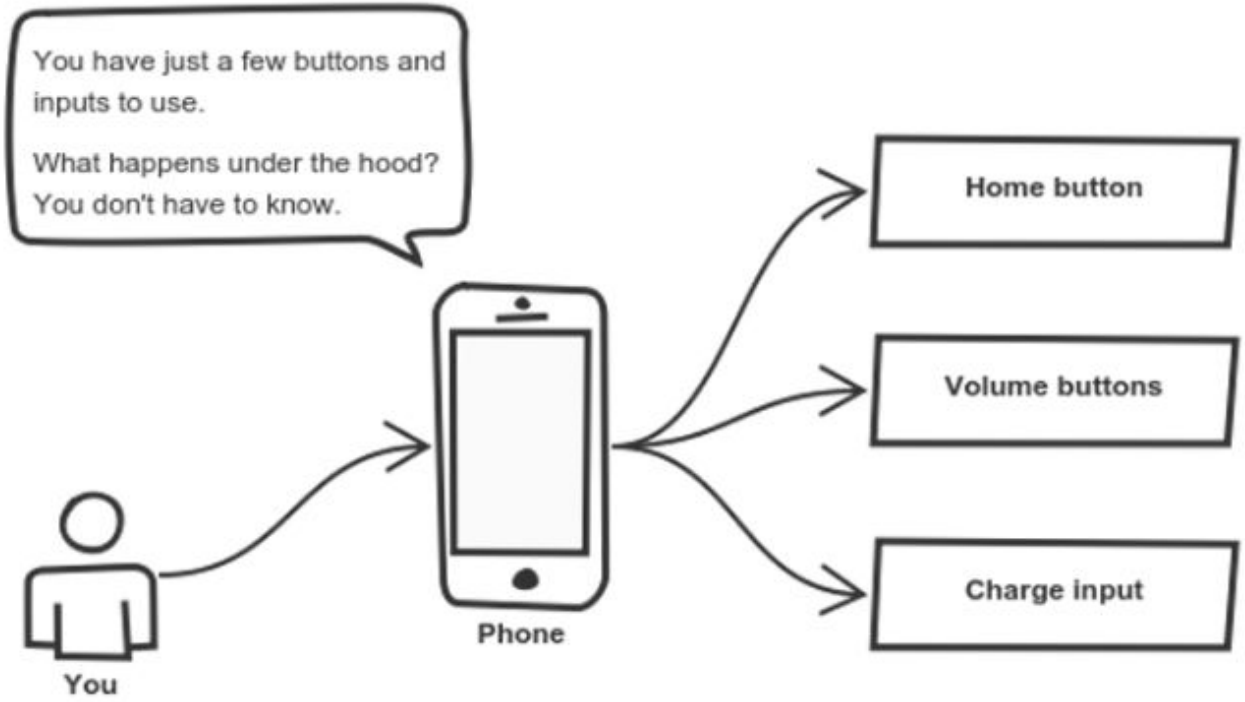
Objects are instances of classes.


Abstraction
Polymorphism
OBJECT
ORIENTED
PROGRAMMING
Encapsulation
Inheritance

Abstraction

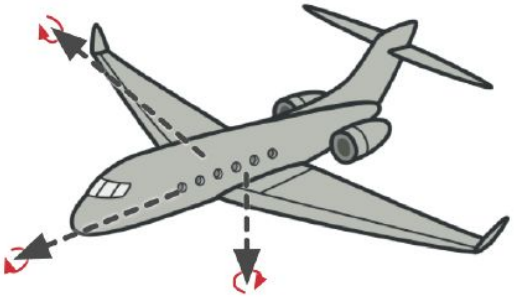
Hides all complicated and unnecessary details from the clients.

Objects only model attributes and behaviors of real objects in a specific context, ignoring the rest.



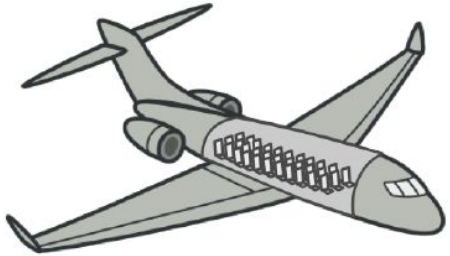
Cellphones are complex, but using them is simple.

Airplane in a flight simulator



Airplane
- speed - altitude - rollAngle - pitchAngle - yawAngle
+ fly()

Airplane in a reservation system



Airplane
- seats
+ reserveSeat(n)

Different models of the same real-world object.

Encapsulation

A protective wrapper to prevent the data and code from being arbitrarily accessed from outside.

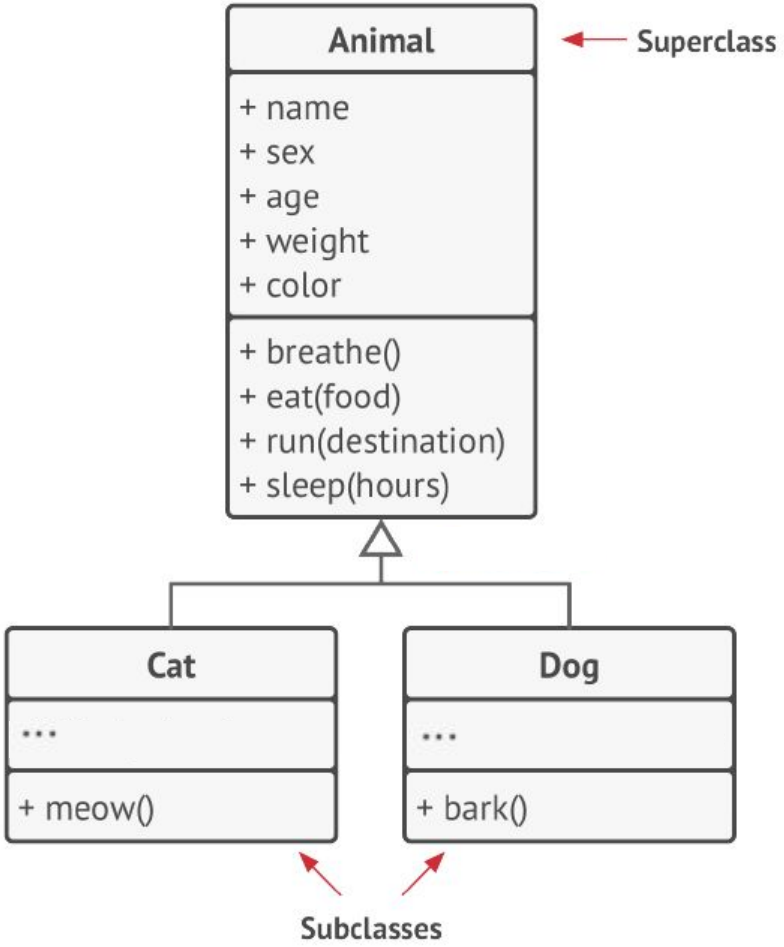
```java
public class AdminLogin {

    // Information might be sensitive
    // Best to hide from outside
    private int passwordHash;

    // A public method to facilitate authentication
    public boolean matchPassword(String password) {
        return password.hashCode() == passwordHash;
    }

}
```

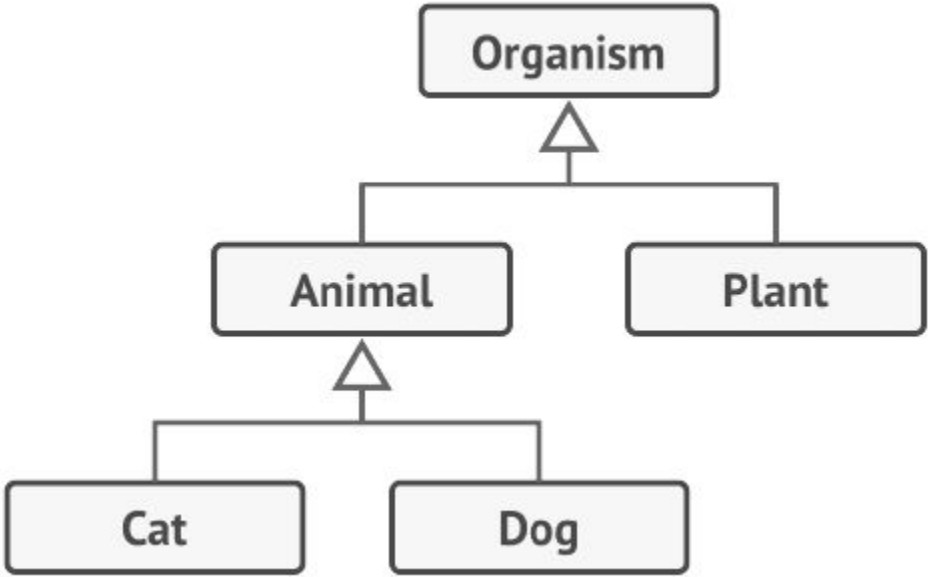
Encapsulation

By default, we keep every member as much encapsulated as possible.

Inheritance

One class uses the properties of other class.

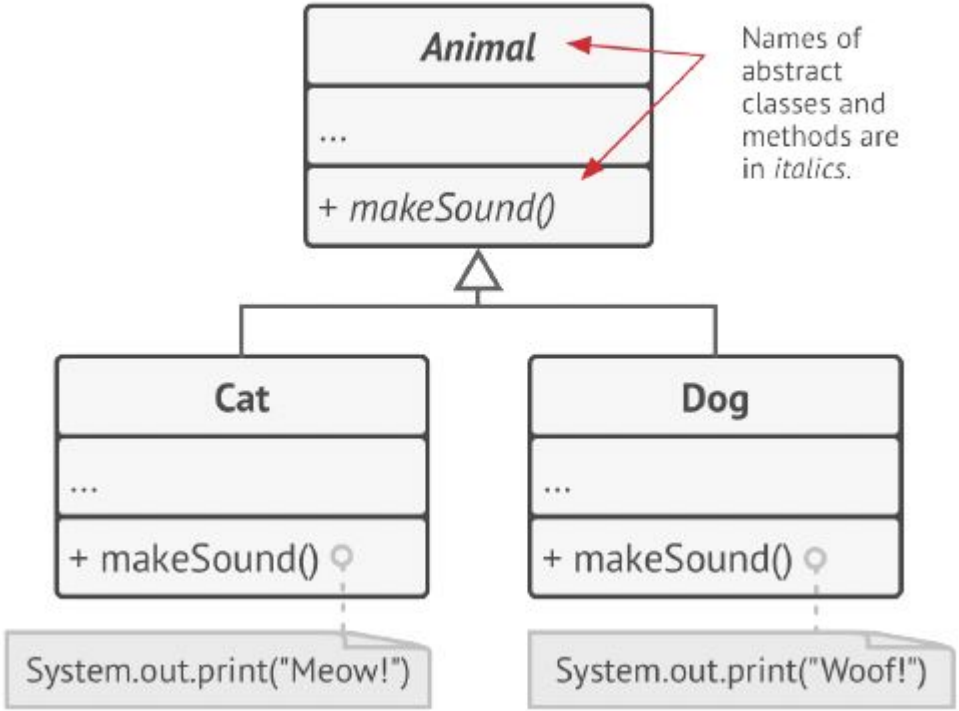

Animal
Superclass
+ name
+ sex
+ age
+ weight
+ color
+ breathe()
+ eat(food)
+ run(destination)
+ sleep(hours)
Cat
...
+ meow()
Dog
...
+ bark()
Subclasses



Multilevel Inheritance

Polymorphism

Allows one common way for some related purposes.



makeSound() has many morphs

Courtesies

1. https://refactoring.guru/design-patterns/book