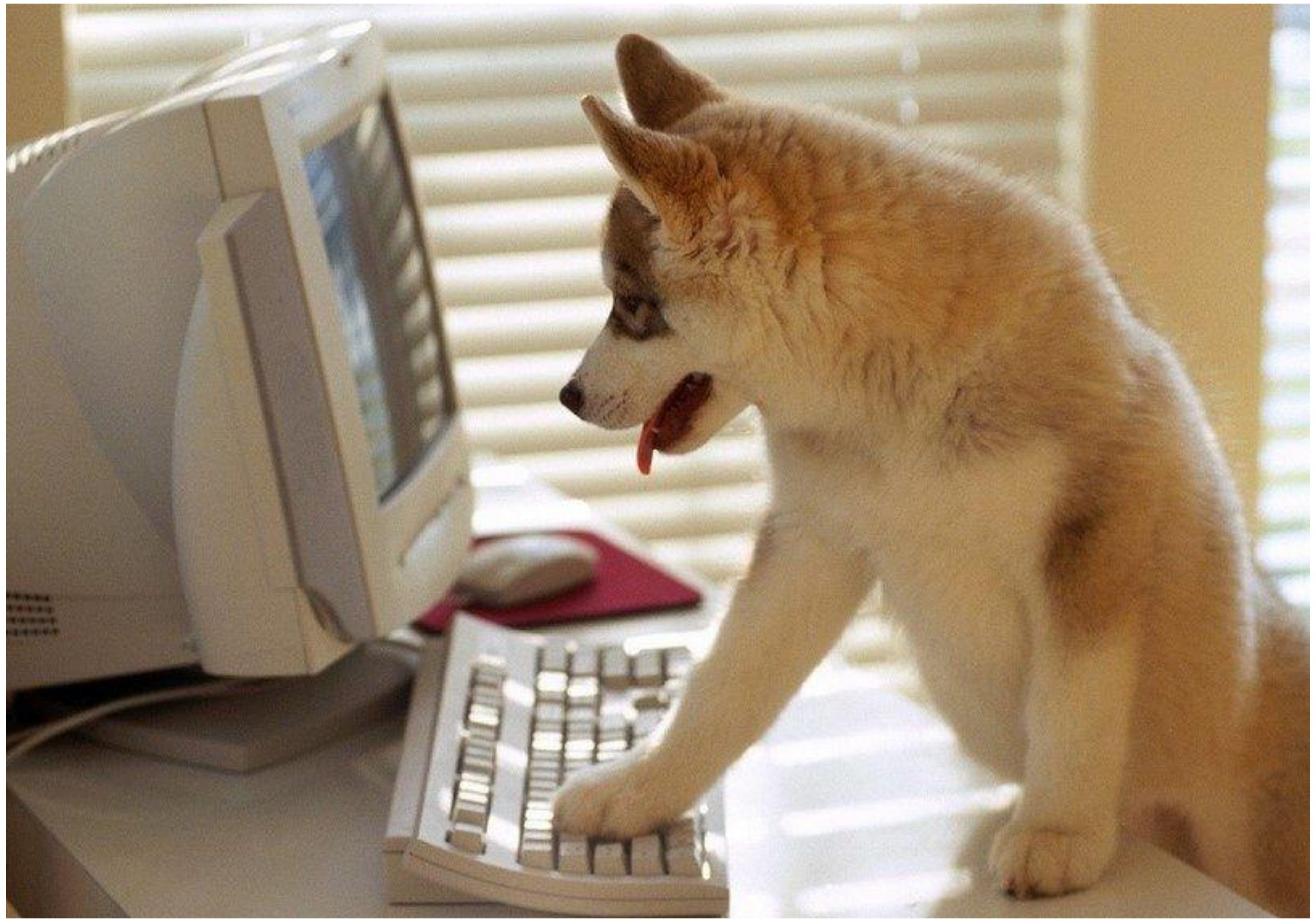
Java Programming



OOP2

```
public abstract class Dog{

   protected String name;

   public abstract void bark();

}
```

OOP2

```
public abstract class Dog{
   protected String name;
   public abstract void bark();
}
```

`abstract` : Declaration without implementation

OOP2

```
public abstract class Dog{

   protected String name;

   public abstract void bark();

}
```

An abstract class is a class with at least one abstract method.

OOP2

```
public abstract class Dog{

   protected String name;

   public abstract void bark();

}
```

```
public class Bulldog extends Dog{

   public void bark(){
      System.out.println(“Woof!”);
   }

}
```

OOP2

```
public interface iWalkable{

   void walk();

}
```

An Interface is like a service, not like a relation.
It contains just constants and abstract methods (no keyword abstract needed)

OOP2

```
public interface iWalkable{

   void walk();

}
```

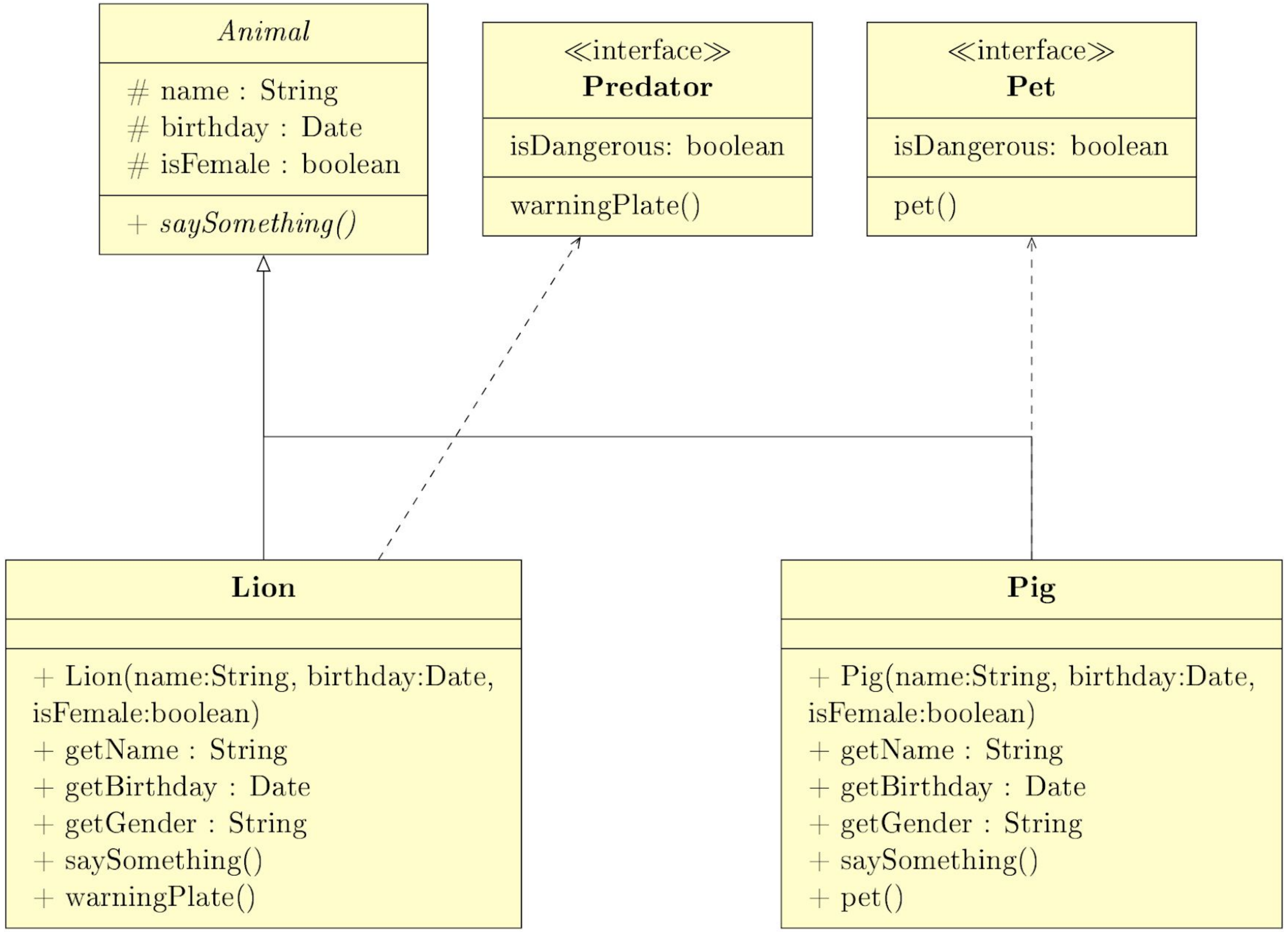
```
public class Bulldog extends Dog implements iWalkable{

   public void bark(){
      System.out.println(“Woof!”);
   }

   public void walk(){
      System.out.println(“I’m walking like a Bulldog”);
   }

}
```

OOP2



Animal

# name : String
# birthday : Date
# isFemale : boolean

+ *saySomething()*

≪interface≫
Predator

isDangerous: boolean

warningPlate()

≪interface≫
Pet

isDangerous: boolean

pet()

Lion

+ Lion(name:String, birthday:Date, isFemale:boolean)
+ getName : String
+ getBirthday : Date
+ getGender : String
+ saySomething()
+ warningPlate()

Pig

+ Pig(name:String, birthday:Date, isFemale:boolean)
+ getName : String
+ getBirthday : Date
+ getGender : String
+ saySomething()
+ pet()

OOP2

Today’s Assignment:
https://classroom.github.com/a/DI-WylpM