COMP90041

Programming and Software Development

Semester 1, 2021

Lab 8

Andrew Naughton

andrew.naughton@unimelb.edu.au

Outline

- Abstract Classes
- Interfaces
- Assignment 2

THE UNIVERSITY OF
MELBOURNE

Abstract Classes

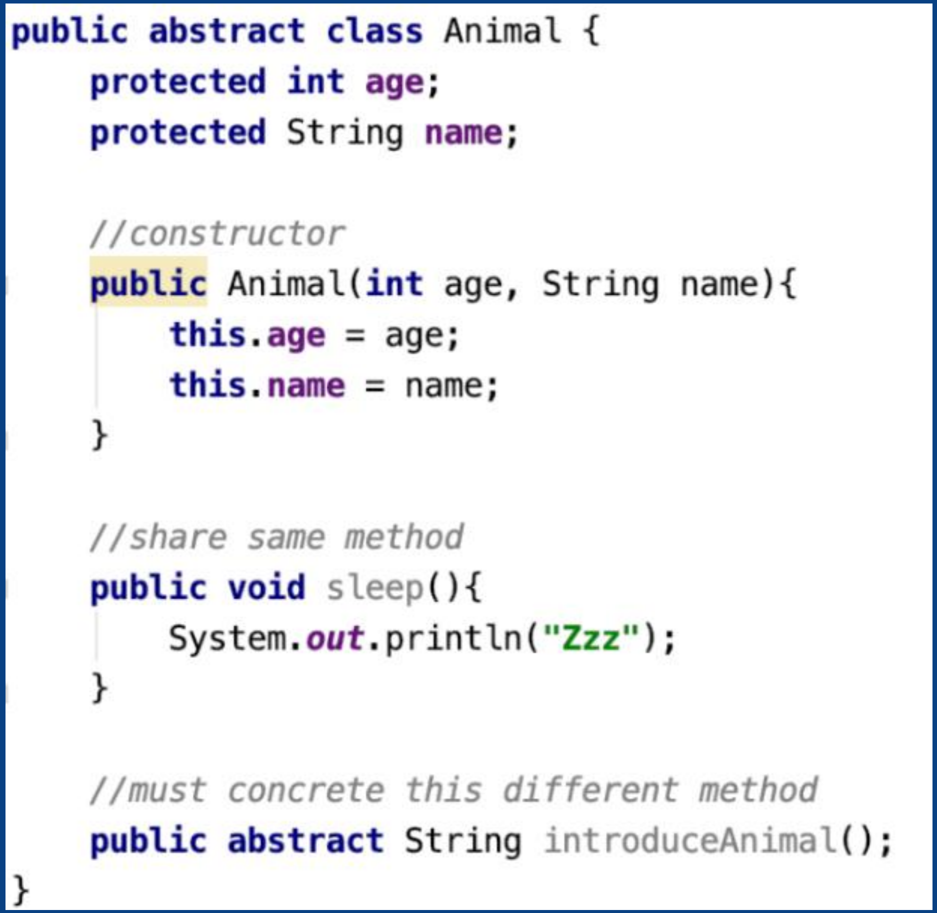
- Of the form:
  - `public abstract class MyAbstract {…}`
- Has abstract methods, which are of the form:
  - `public abstract type methodName(params);`
- To *access* an abstract class:
  - `public class MyClass extends MyAbstract {…}`

Abstract Classes

- Purpose: to allow a number of closely related classes to implement common methods

- Note:
  - Cannot create an instance of an abstract class
  - A class with abstract methods must be declared as abstract
  - Any class that extends an abstract class must implement (override) all of the abstract class' abstract methods

THE UNIVERSITY OF MELBOURNE

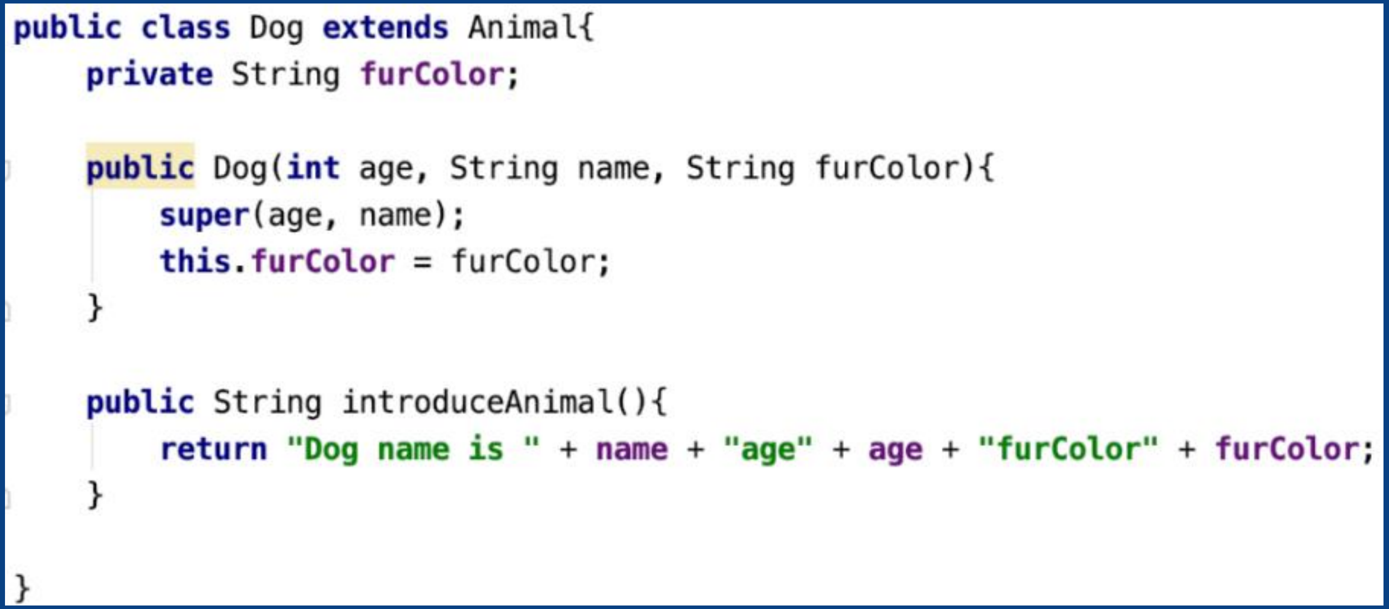


```java
public abstract class Animal {
    protected int age;
    protected String name;

    //constructor
    public Animal(int age, String name){
        this.age = age;
        this.name = name;
    }

    //share same method
    public void sleep(){
        System.out.println("Zzz");
    }

    //must concrete this different method
    public abstract String introduceAnimal();
}
```

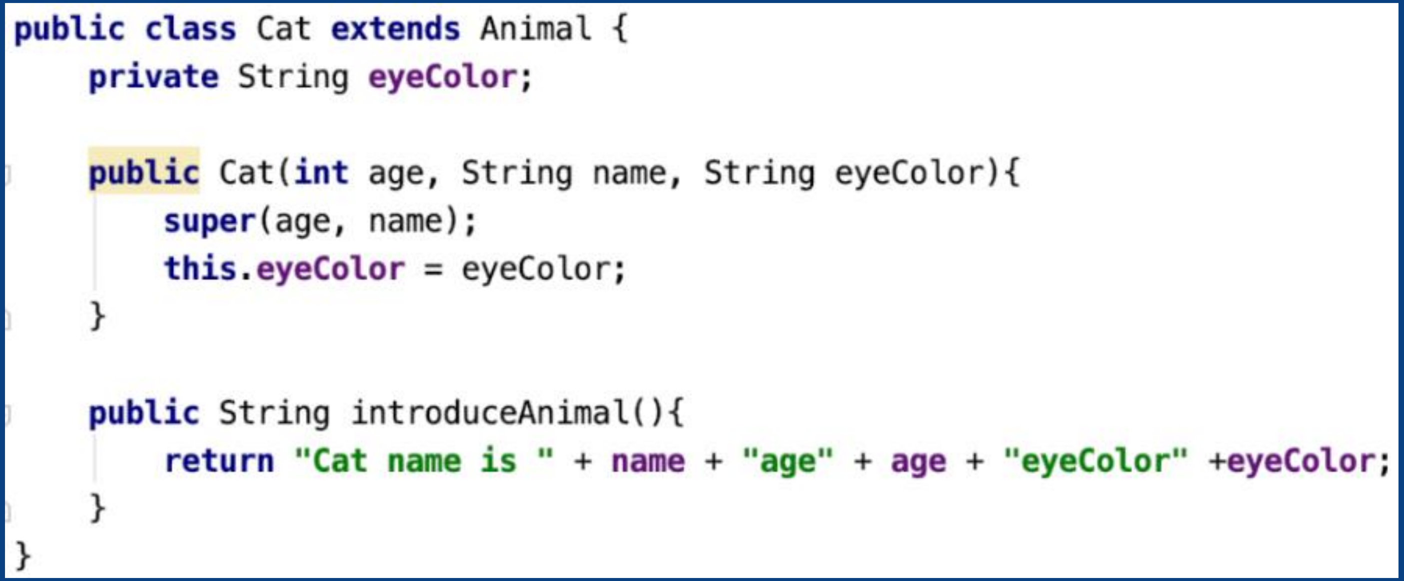


```java
public class Dog extends Animal{
    private String furColor;

    public Dog(int age, String name, String furColor){
        super(age, name);
        this.furColor = furColor;
    }

    public String introduceAnimal(){
        return "Dog name is " + name + "age" + age + "furColor" + furColor;
    }

}
```



```java
public class Cat extends Animal {
    private String eyeColor;

    public Cat(int age, String name, String eyeColor){
        super(age, name);
        this.eyeColor = eyeColor;
    }

    public String introduceAnimal(){
        return "Cat name is " + name + "age" + age + "eyeColor" +eyeColor;
    }
}
```

Interfaces

- Of the form:
  - `public interface MyInterface {…}`

- To *access* an interface:
  - `public class MyClass implements MyInterface {…}`

- To *access* more than one interface: (use commas)
  - `public class MyClass implements Ifirst, ISecond{…}`

THE UNIVERSITY OF
MELBOURNE

Interfaces

- Purpose: To allow unrelated classes to implement common methods

- Like abstract classes, cannot create instances of an interface
- No constructor(s)
- More abstract than an abstract class
  - Cannot have instance variables / methods
  - Typically just has abstract methods

THE UNIVERSITY OF MELBOURNE

Assignment 2

- Spec
- Questions