OO Programming

Presented by

Dr Rubeena Doomun

Class

- A class is defined as a template/ blue print that describes the behaviours/states that an object of its type support.
- It describes a group of objects that exhibit similar properties (attributes) and common behaviour (methods).
- Example:

  Class Car

  Attributes: Colour, Model, Cost

  Methods: go(), stop()

Object

- An object is an instance of a class. It has states and behaviours.
- Example:

  A dog has states – colour white, name Bob, breed Labrador

  A dog has behaviours – barking, eating

Message Passing

- Objects interact and communicate with each other using messages.
- An object asks another object to perform an action by sending it a *message.*
- Rather than allowing an object direct access to another's object data, a message is sent to the target object requesting information.

3 steps when creating an object from a class

There are three steps when creating an object from a class:

- **Declaration:** A variable declaration with a variable name with an object type.
- **Instantiation:** The 'new' keyword is used to create the object.
- **Initialization:** The 'new' keyword is followed by a call to a constructor. This call initializes the new object.

Controlling Access to Members of a Class

- Access level modifiers determine whether other classes can use a particular field or invoke a particular method.

There are 3 types of access control:

Public can be accessed outside the class.

Private can be accessed within the class.

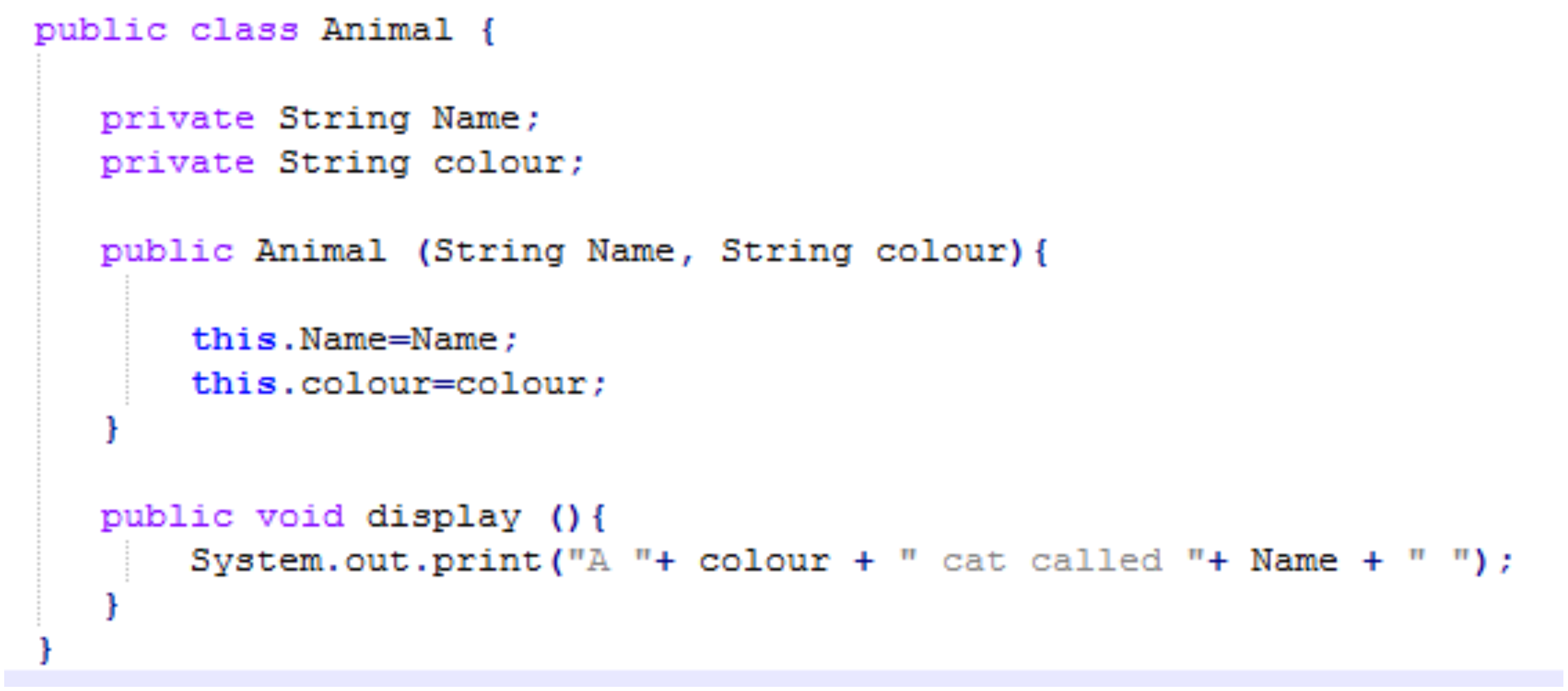
Protected can be accessed within the class and by subclasses.

Constructor

- A **constructor in Java** is a block of code similar to a method that's called when an instance of an object is created.
- Example:

```
Animal(){
}
```

Example



```java
public class Animal {

    private String Name;
    private String colour;

    public Animal (String Name, String colour){

        this.Name=Name;
        this.colour=colour;
    }

    public void display (){
        System.out.print("A "+ colour + " cat called "+ Name + " ");
    }
}
```

Data abstraction

- The idea of abstraction is to hide away complex low level details so that we can concentrate on essential characteristics more easily
- In OO paradigm, classes are used to implement data abstraction
    - Hide complex details about data and algorithm
    - Provide a simple interface for programmers to use
    - Example: Steering Wheel

Encapsulation

- Encapsulation is a process of grouping together all the data and operations that relate to a single concept, usually grouping them together in a single syntactic unit.
- Example: human body
- Encapsulation refers to the state of objects - objects encapsulate their state and hide it from the outside; outside users of the class interact with it through its methods, but cannot access the classes state directly.

Polymorphism

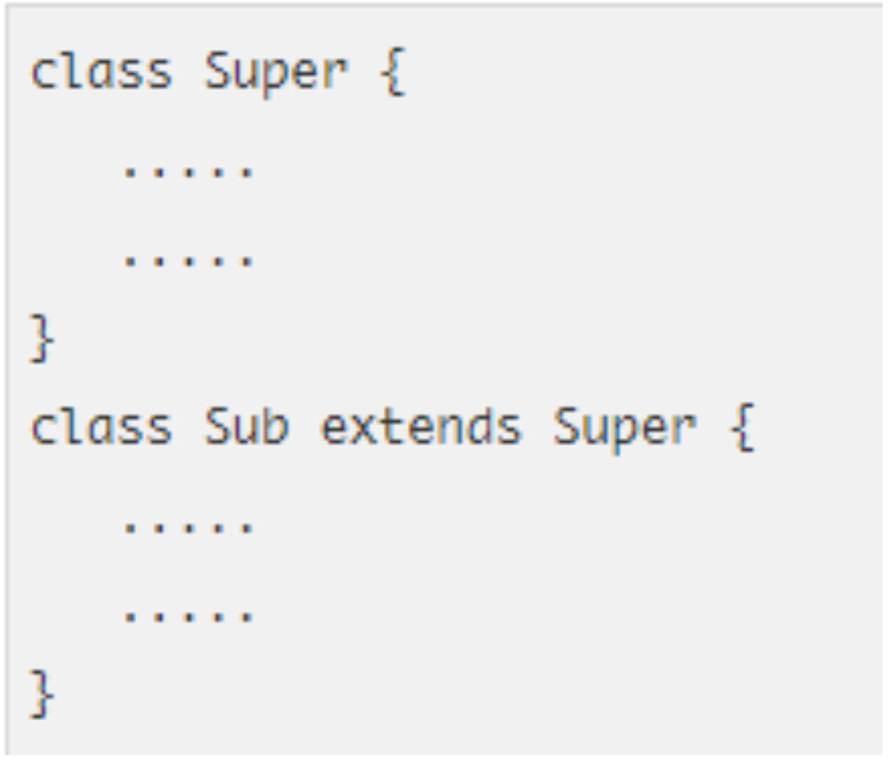
- In the OO context, polymorphism means objects can take on or assume many different forms.
- The same operation can behave differently on different classes. Or simply, different objects can respond to the same message in different ways.
- Polymorphism allows many methods of a class to have the same name.

Inheritance

- Inheritance can be defined as the process where one class acquires the properties (methods and fields) of another.
- With the use of inheritance the information is made manageable in a hierarchical order.
- The class which inherits the properties of other is known as subclass (derived class, child class) and the class whose properties are inherited is known as superclass (base class, parent class).

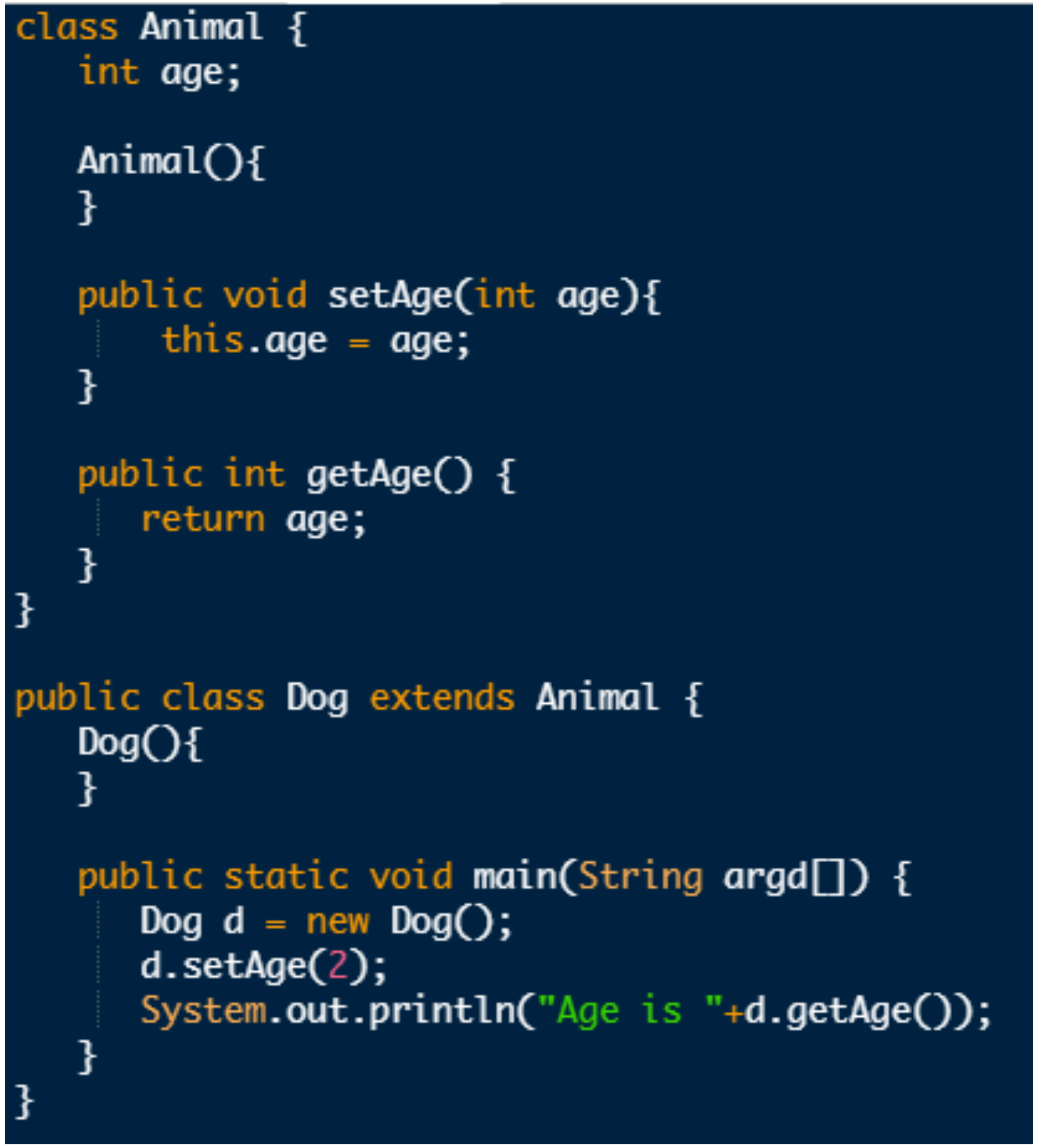
extends Keyword

- **extends** is the keyword used to inherit the properties of a class. Following is the syntax of extends keyword.



```
class Super {
   .....
   .....
}
class Sub extends Super {
   .....
   .....
}
```

Example



```
class Animal {
   int age;

   Animal(){
   }

   public void setAge(int age){
       this.age = age;
   }

   public int getAge() {
      return age;
   }
}

public class Dog extends Animal {
   Dog(){
   }

   public static void main(String argd[]) {
      Dog d = new Dog();
      d.setAge(2);
      System.out.println("Age is "+d.getAge());
   }
}
```

References

- https://docs.oracle.com/javase/tutorial/java/javaOO/accesscontrol.html
- https://www.tutorialspoint.com/java/java_inheritance.htm