INTRODUCTION TO PROGRAMMING LANGUAGE II(JAVA)

Fariha Zahin
Lecturer
CSE, Southeast University

Object Oriented Programming in JAVA

What is Inheritance?

•Inheritance is a mechanism in Java where one class acquires the properties and behaviors (fields and methods) of another class.

•The class that inherits is called **Subclass (Child class)**.

•The class being inherited from is called **Superclass (Parent class)**.

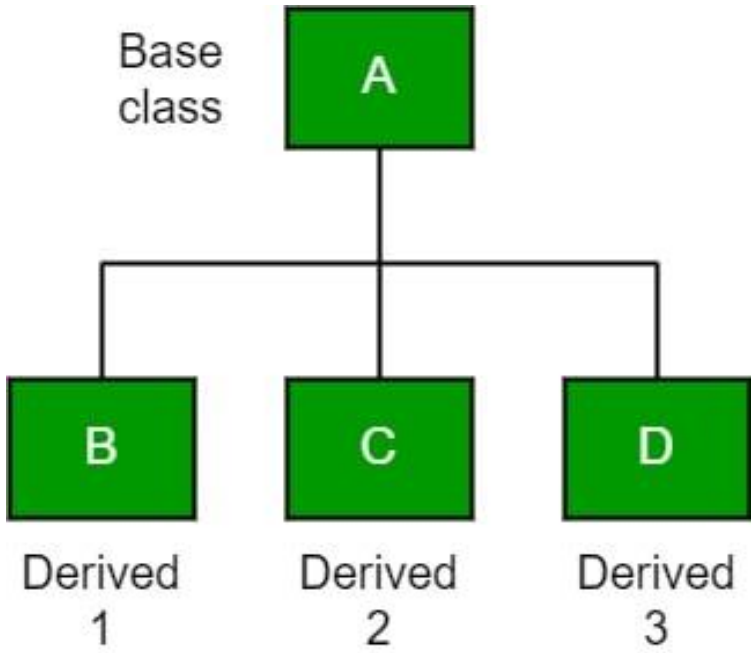
Why Use Inheritance?

•**Code Reusability**: Reuse existing code from the superclass.

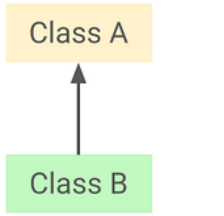
•**Method Overriding**: Modify superclass methods in subclass.

•**Runtime Polymorphism**: Enables dynamic method dispatch.

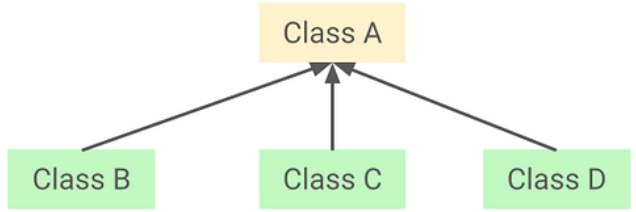
•**Organizes Code**: Establishes a natural hierarchy.



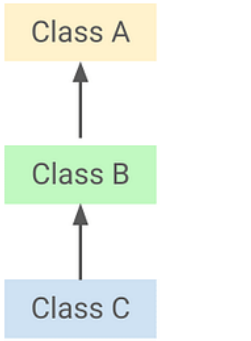
Types of Inheritance in Java
1.Single Inheritance
2.Multilevel Inheritance
3.Hierarchical Inheritance (Note: Java does not support multiple inheritance through classes. It is achieved using interfaces.)



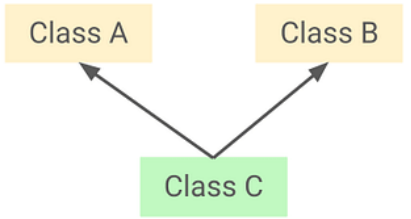
Single Inheritance



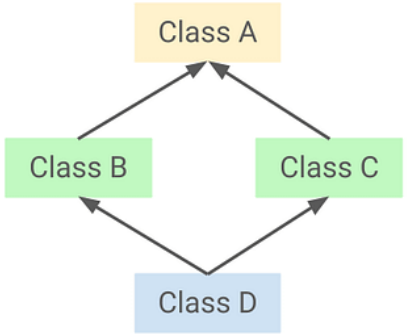
Hierarchical inheritance



Multilevel Inheritance



Multiple Inheritance



Hybrid Inheritance

Java uses the extends keyword to establish an inheritance relationship between a subclass and a superclass.
Syntax:

```
class Subclass extends Superclass {
    // body of subclass
}
```

- The subclass inherits accessible fields and methods from the superclass.
- It allows the subclass to reuse, override, or add new functionality.

Single Inheritance

- A subclass inherits from only one superclass.
- Promotes code reuse and method overriding.

Main.java

```java
class Animal {  1 usage  1 inheritor
    void eat() {  1 usage
        System.out.println("This animal eats food.");
    }
}

class Dog extends Animal {  2 usages
    void bark() {  1 usage
        System.out.println("The dog barks.");
    }
}

public class Main {
    public static void main(String[] args) {
        Dog d = new Dog();
        d.eat(); // Inherited method
        d.bark(); // Own method
    }
}
```

Run — Main

```
C:\Users\u1904\.jdks\openjdk-25\bin\java.exe "-javaa
This animal eats food.
The dog barks.

Process finished with exit code 0
```

Multilevel Inheritance

- A class is derived from another derived class, forming a chain of inheritance.
- Supports multi-level code reuse.

Main.java

```java
class Animal {  1 usage  2 inheritors
    void eat() {  1 usage
        System.out.println("This animal eats food.");
    }
}
class Dog extends Animal {  1 usage  1 inheritor
    void bark() {  1 usage
        System.out.println("The dog barks.");
    }
}
class Puppy extends Dog {  2 usages
    void weep() {  1 usage
        System.out.println("The puppy weeps.");
    }
}
public class Main {
    public static void main(String[] args) {
        Puppy p = new Puppy();
        p.eat();   // Grandparent class (Animal)
        p.bark();  // Parent class (Dog)
        p.weep();  // Own method (Puppy)
    }
}
```

Run Main

```
C:\Users\u1904\.jdks\openjdk-25\bin\java.exe "-javaagen
This animal eats food.
The dog barks.
The puppy weeps.

Process finished with exit code 0
```

Hierarchical Inheritance

- Multiple subclasses inherit from a single superclass.
- Promotes a hierarchical classification.

```
C:\Users\u1904\.jdks\openjdk-25\bin\ja
This animal eats food.
The dog barks.
This animal eats food.
The dog barks.
The puppy weeps.
This animal eats food.
The cat meows.

Process finished with exit code 0
```

Main.java

```java
class Dog extends Animal {  3 usages  1 inheritor
        System.out.println("The dog barks.");
    }
}
class Puppy extends Dog {  2 usages
    void weep() {  1 usage
        System.out.println("The puppy weeps.");
    }
}
class Cat extends Animal {  2 usages
    void meow() {  1 usage
        System.out.println("The cat meows.");
    }
}
public class Main {
    public static void main(String[] args) {
        Dog d = new Dog();
        d.eat();     // Inherited from Animal
        d.bark();    // Own method

        Puppy p = new Puppy();
        p.eat();     // Grandparent (Animal)
        p.bark();    // Parent (Dog)
        p.weep();    // Own method

        Cat c = new Cat();
        c.eat();     // Inherited from Animal
        c.meow();    // Own method
    }
}
```