Java Tutorial

Inheritance

```
class Animal {

  // field and method of the parent class
  String name;
  public void eat() {
    System.out.println("I can eat");
  }
}

// inherit from Animal
class Dog extends Animal {

  // new method in subclass
  public void display() {
    System.out.println("My name is " + name);
  }
}

class Main {
  public static void main(String[] args) {

    // create an object of the subclass
    Dog labrador = new Dog();

    // access field of superclass
    labrador.name = "Rohu";
    labrador.display();

    // call method of superclass
    // using object of subclass
    labrador.eat();

  }
}
```

Method Overriding in Java

```
class Animal {

  // method in the superclass
  public void eat() {
    System.out.println("I can eat");
  }
}
```

```
// Dog inherits Animal
class Dog extends Animal {

  // overriding the eat() method
  @Override
  public void eat() {
    System.out.println("I eat dog food");
  }

  // new method in subclass
  public void bark() {
    System.out.println("I can bark");
  }
}

class Main {
  public static void main(String[] args) {

    // create an object of the subclass
    Dog labrador = new Dog();

    // call the eat() method
    labrador.eat();
    labrador.bark();
  }
}
```

Supper Keyword in Java Inheritance

```
class Animal {

  // method in the superclass
  public void eat() {
    System.out.println("I can eat");
  }
}

// Dog inherits Animal
class Dog extends Animal {

  // overriding the eat() method
  @Override
  public void eat() {

    // call method of superclass
    super.eat();
    System.out.println("I eat dog food");
  }
```

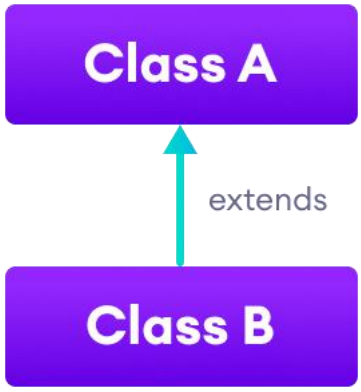
```
  // new method in subclass
  public void bark() {
    System.out.println("I can bark");
  }
}

class Main {
  public static void main(String[] args) {

    // create an object of the subclass
    Dog labrador = new Dog();

    // call the eat() method
    labrador.eat();
    labrador.bark();
  }
}
```

Accessing Protected methods and properties in inheritance

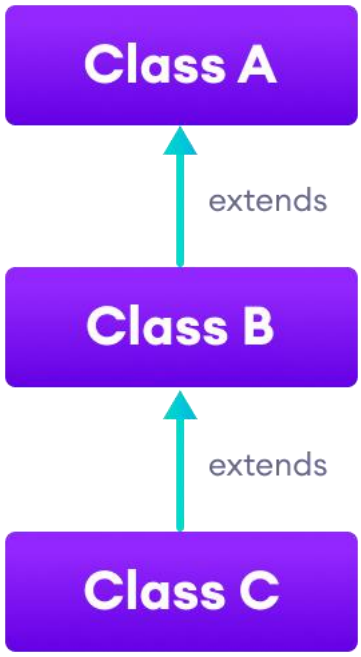
```
class Animal {
  protected String name;

  protected void display() {
    System.out.println("I am an animal.");
  }
}

class Dog extends Animal {

  public void getInfo() {
    System.out.println("My name is " + name);
  }
}

class Main {
  public static void main(String[] args) {

    // create an object of the subclass
    Dog labrador = new Dog();

    // access protected field and method
    // using the object of subclass
    labrador.name = "Rocky";
    labrador.display();

    labrador.getInfo();
  }
}
```

Type of Inheritance

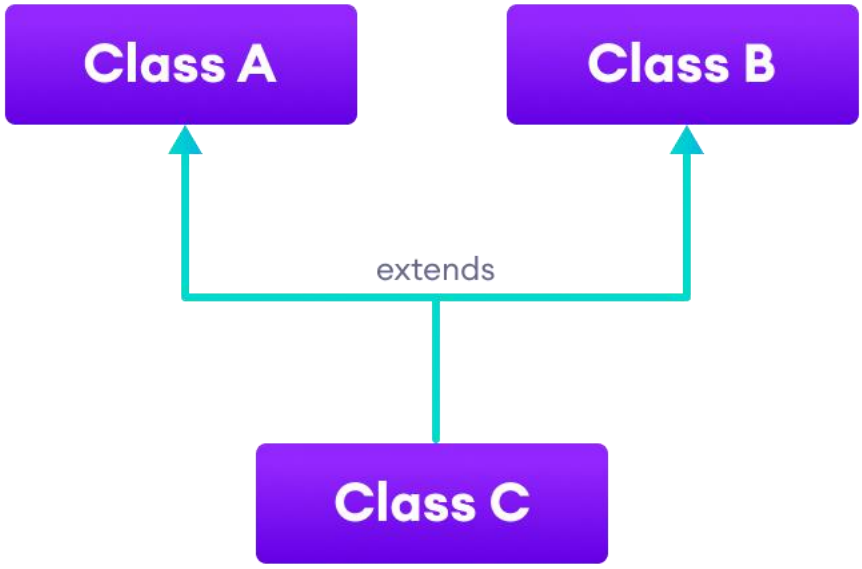
1. Single Inheritance



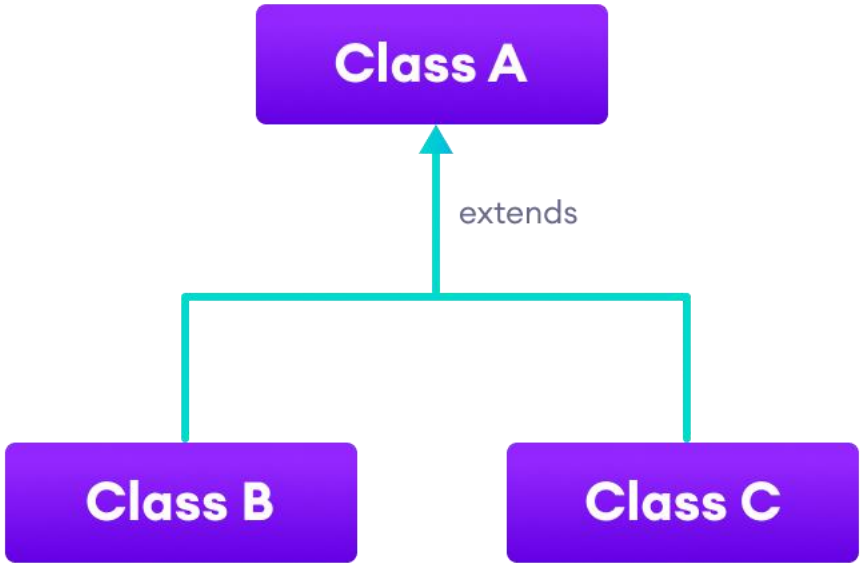
2. Multi Level Inheritance



3. Multiple Inheritance



4. Hierarchical Inheritance



Access Modifiers

1. Private

```
class Data {
   private String name;

   // getter method
   public String getName() {
      return this.name;
   }
   // setter method
   public void setName(String name) {
      this.name= name;
   }
}
public class Main {
   public static void main(String[] main){
      Data d = new Data();

      // access the private variable using the getter and setter
      d.setName("Programiz");
      System.out.println(d.getName());
   }
}
```

2. Protected

```
class Animal {
   // protected method
   protected void display() {
      System.out.println("I am an animal");
   }
}

class Dog extends Animal {
   public static void main(String[] args) {

      // create an object of Dog class
      Dog dog = new Dog();
       // access protected method
      dog.display();
   }
}
```

3. Public

```
// Animal.java file
// public class
public class Animal {
```

```
    // public variable
    public int legCount;

    // public method
    public void display() {
        System.out.println("I am an animal.");
        System.out.println("I have " + legCount + " legs.");
    }
}

// Main.java
public class Main {
    public static void main( String[] args ) {
        // accessing the public class
        Animal animal = new Animal();

        // accessing the public variable
        animal.legCount = 4;
        // accessing the public method
        animal.display();
    }
}
```

4. Default

```
package defaultPackage;
class Logger {
    void message(){
        System.out.println("This is a message");
    }
}
```