Instance variable

↳ These are declared in class but outside a method constructor or any block

↳ Instance variable are created when an object is created with the use of the keyword 'new' & destroyed when the object is destroyed.

↳ These are declared at the class level before or after use.

↳ These variables are visible for all methods, constructors, & blocks in the class.

↳ Instance variable have default values

For numbers —— 0

For Booleans —— f

For object Reference —— NULL

values can be assigned during the declaration or within the constructor.

↳ These can be accessed directly by calling the variable name inside the class.
However within static methods (when instance variable are given accessibility) they should be called using the fully qualified name.

object reference. variable name.

INHERITANCE

↳ Inheritance is one of the key feature of OOPs that allows us to create a new class from an existing class.

↳ The new class that is created is known as Subclass (child or derived class) and the existing class from where the child class is derived is known as Superclass (parent or base class).

↳ The "extends" keyword is used to perform inheritance in java

↳ For example:

```
Class animal{
    // methods & fields
}

// use of extend keyword to perform inheritance
Class dog extends animal{
    // methods & fields of animals
    // methods & fields of class
}
```

"Syntax"

```
Class derived-class extends base-class
{
    // methods & functions
}
```

```
class animal {
  // field & method of 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 labuadou = new dog();
  // access fields of superclass
    labuadou.name = "Rohu";
    labuadou.display();
  // call method of superclass using object of subclass
    labuadou.eat();
  }
  }
```

```
My name is Rohu
I can eat
```

Important terminology

(a) Super Class -

The class whose features are inherited is known as superclass (or base or parent class).

(b) Sub class -

↳ The class that inherits the other class is known as sub class (or derived class, extended class, child class)

↳ It can add its own fields & methods in addition to the superclass fields & methods

(c) Reusability -

↳ It supports the concept of "reusability"

↳ When we want to create a new class & there is already a class that includes some of the code that we want, we can derive our new class from the existing class.

↳ By doing this we are reusing the fields & methods of the existing class.

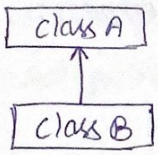
ct of subclass

Types of Inheritance

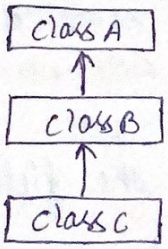
↳ Three types of inheritance in java
- (a) Single
- (b) Multilevel
- (c) Hierarchical

↳ Multiple & hybrid inheritance is supported through "interface" only.

(a) Single Inheritance

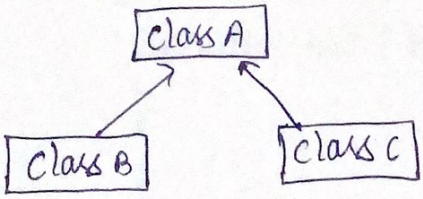


(b) Multilevel Inheritance



When there is a chain of inheritance, it is known as multilevel inheritance.

(c) Hierarchical Inheritance



When two or more classes inherits a single class it is known as hierarchical inheritance.

* Why multiple
↳ To avoid multiple
↳ Conside
↳
Dac

* Why multiple inheritance is not supported in java

↳ To reduce the complexity & simplify the language, multiple inheritance is not supported in java.

↳ Consider a scenario where A, B, & C are three classes

↳ The C class inherits A & B classes

↳ A & B class have the same method & you call it from a child class object there will be ambiguity to call the method of A & B class.

↳ Java renders compile-time error if you inherit 2 classes. So whether you have same method or different, there will be compile time error.

ted through

of inheritance,
inheritance.

classes
s it is
al inheritance.

Facts about Inheritance