**Inheritance in Java**

The process of obtaining the data members and methods from one class to another class is known as **inheritance**. It is one of the fundamental features of object-oriented programming.

**Important points**

* In the inheritance the class which is give data members and methods is known as base or super or parent class.
* The class which is taking the data members and methods is known as sub or derived or child class.
* The data members and methods of a class are known as features.
* The concept of inheritance is also known as re-usability or extendable classes or sub classing or derivation.

**Why use Inheritance ?**

* For Method Overriding (used for Runtime Polymorphism).
* It's main uses are to enable polymorphism and to be able to reuse code for different classes by putting it in a common super class
* For code Re-usability

**Syntax of Inheritance**

**class** Subclass-Name **extends** Superclass-Name

{

//methods and fields

}

**Real life example of inheritance**

The real life example of inheritance is child and parents, all the properties of father are inherited by his son.

**Advantage of inheritance**

If we develop any application using concept of Inheritance than that application have following advantages,

* Application development time is less.
* Application take less memory.
* Application execution time is less.
* Application performance is enhance (improved).
* Redundancy (repetition) of the code is reduced or minimized so that we get consistence results and less storage cost.

**Note:**In Inheritance the scope of access modifier increasing is allow but decreasing is not allow. Suppose in parent class method access modifier is default then it's present in child class with default or public or protected access modifier but not private(it decreased scope).

**Tpyes of Inheritance**

Based on number of ways inheriting the feature of base class into derived class we have five types of inheritance; they are:

* Single inheritance
* Multiple inheritance
* Hierarchical inheritance
* Multilevel inheritance
* Hybrid inheritance

**Single inheritance**

In single inheritance there exists single base class and single derived class.

**Example of Single Inheritance**

**class** Faculty

{

**float** salary=30000;

}

**class** Science **extends** Faculty

{

**float** bonous=2000;

**public** **static** **void** main(String args[])

{

Science obj=**new** Science();

System.**out**.println("Salary is:"+obj.salary);

System.**out**.println("Bonous is:"+obj.bonous);

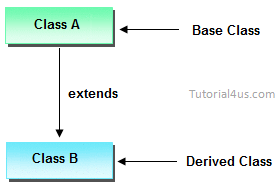
}

}

**Output**

Salary is: 30000.0

Bonous is: 2000.0



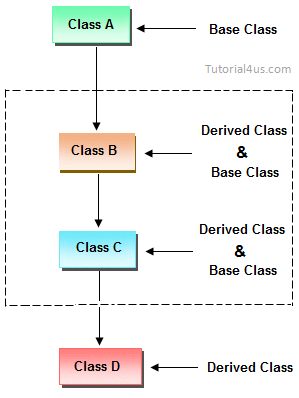
**Multilevel inheritances in Java**

In Multilevel inheritances there exists single base class, single derived class and multiple intermediate base classes.

**Single base class + single derived class + multiple intermediate base classes.**

**Intermediate base classes**

An intermediate base class is one in one context with access derived class and in another context same class access base class.



Hence all the above three inheritance types are supported by both classes and interfaces.

**Example of Multilevel Inheritance**

**class** Faculty

{

**float** total\_sal=0, salary=30000;

}

**class** HRA **extends** Faculty

{

**float** hra=3000;

}

**class** DA **extends** HRA

{

**float** da=2000;

}

**class** Science **extends** DA

{

**float** bonous=2000;

**public** **static** **void** main(String args[])

{

Science obj=**new** Science();

obj.total\_sal=obj.salary+obj.hra+obj.da+obj.bonous;

System.**out**.println("Total Salary is:"+obj.total\_sal);

}

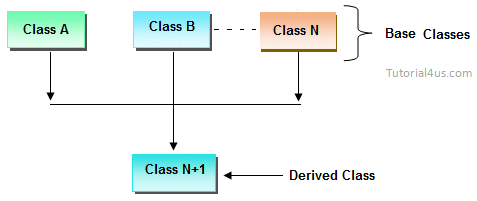
}

**Output**

Total Salary is: 37000.0

**Multiple inheritance**

In multiple inheritance there exist multiple classes and singel derived class.

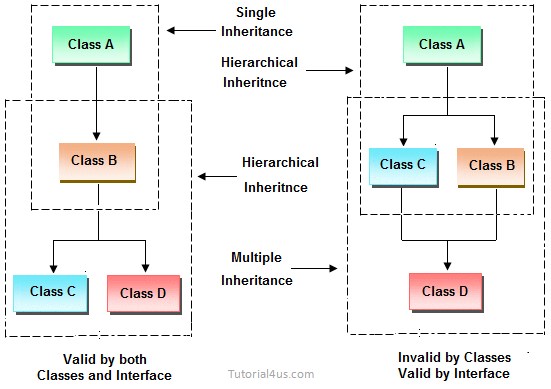


The concept of multiple inheritance is not supported in java through concept of classes but it can be supported through the concept of interface.

**Hybrid inheritance**

Combination of any inheritance type

In the combination if one of the combination is multiple inheritance then the inherited combination is not supported by java through the classes concept but it can be supported through the concept of interface.



**Inheriting the feature from base class to derived class**

In order to inherit the feature of base class into derived class we use the following syntax

**Syntax**

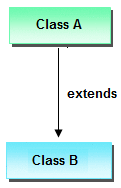
**class** ClassName-2 **extends** ClasssName-1

{

variable declaration;

Method declaration;

}



**Explanation**

1. ClassName-1 and ClassName-2 represents name of the base and derived classes respectively.
2. extends is one of the keyword used for inheriting the features of base class into derived class it improves the functionality of derived class.

**Important Points for Inheritance:**

* In java programming one derived class can extends only one base class because java programming does not support multiple inheritance through the concept of classes, but it can be supported through the concept of Interface.
* Whenever we develop any inheritance application first create an object of bottom most derived class but not for top most base class.
* When we create an object of bottom most derived class, first we get the memory space for the data members of top most base class, and then we get the memory space for data member of other bottom most derived class.
* Bottom most derived class contains logical appearance for the data members of all top most base classes.
* If we do not want to give the features of base class to the derived class then the definition of the base class must be preceded by final hence final base classes are not reusable or not inheritable.
* If we are do not want to give some of the features of base class to derived class than such features of base class must be as private hence private features of base class are not inheritable or accessible in derived class.
* Data members and methods of a base class can be inherited into the derived class but constructors of base class can not be inherited because every constructor of a class is made for initializing its own data members but not made for initializing the data members of other classes.
* An object of base class can contain details about features of same class but an object of base class never contains the details about special features of its derived class (this concept is known as scope of base class object).
* For each and every class in java there exists an implicit predefined super class called java.lang.Object. because it providers garbage collection facilities to its sub classes for collecting un-used memory space and improved the performance of java application.

**Example of Inheritance**

**class** Faculty

{

**float** salary=30000;

}

**class** Science **extends** Faculty

{

**float** bonous=2000;

**public** **static** **void** main(String args[])

{

Science obj=**new** Science();

System.**out**.println("Salary is:"+obj.salary);

System.**out**.println("Bonous is:"+obj.bonous);

}

}

**Output**

Salary is: 30000.0

Bonous is: 2000.0

**Why multiple inheritance is not supported in java?**

Due to ambiguity problem java does not support multiple inheritance at class level.

**Example**

**class** A

{

**void** disp()

{

System.**out**.println("Hello");

}

}

**class** B

{

**void** disp()

System.**out**.println("How are you ?");

}

}

**class** C **extends** A,B //suppose if it were

{

Public Static **void** main(String args[])

{

C obj=**new** C();

obj.disp();//Now which disp() method would be invoked?

}

}

In above code we call both class A and class B disp() method then it confusion which class method is call. So due to this ambiguity problem in java do not use multiple inheritance at class level, but it support at interface level.

**Difference between Java Inheritance and C++ Inheritance**

The main difference between java Inheritance and C++ Inheritance is; Java doesn’t support multiple inheritance but C++ support.