**Inheritance in Java**

The process by which one class acquires the properties(data members) and functionalities(methods) of another class is called **inheritance**. The aim of inheritance is to provide the reusability of the code

Inheritance defines is-a relationship between a Super class and its Sub class. extends and implements keywords are used to describe inheritance in Java.

In this example, we have a base class Teacher and a sub class Physics Teacher. Since class Physics Teacher extends the designation and college properties and work () method from base class, we need not to declare these properties and method in sub class.  
Here we have college Name, designation and work () method which are common to all the teachers so we have declared them in the base class, this way the child classes like Math Teacher, Music Teacher and Physics Teacher do not need to write this code and can be used directly from base class.

Based on the above example we can say that Physics Teacher **IS-A** Teacher.

**Private member in parent class**

The derived class inherits all the members and methods that are declared as public or protected. If the members or methods of super class are declared as private then the derived class cannot use them directly. The private members can be accessed only in its own class. Such private members can only be accessed using public or protected getter and setter methods of super class

class Teacher {

private String designation = "Teacher";

private String collegeName = "Beginnersbook";

public String getDesignation() {

return designation;

}

protected void setDesignation(String designation) {

this.designation = designation;

}

protected String getCollegeName() {

return collegeName;

}

protected void setCollegeName(String collegeName) {

this.collegeName = collegeName;

}

void does(){

System.out.println("Teaching");

}

}

public class JavaExample extends Teacher{

String mainSubject = "Physics";

public static void main(String args[]){

JavaExample obj = new JavaExample();

System.out.println(obj.getCollegeName());

System.out.println(obj.getDesignation());

System.out.println(obj.mainSubject);

obj.does();

}

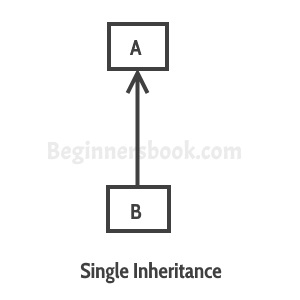
}

If we declared a static variable as private then that variable won’t inherited to the sub classes.

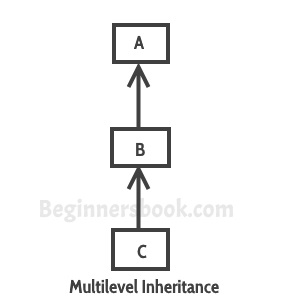
If we declared parent class members as private then we need to create getters and setters so that child can use them.

## **Types of inheritance**

### **Single Inheritance**

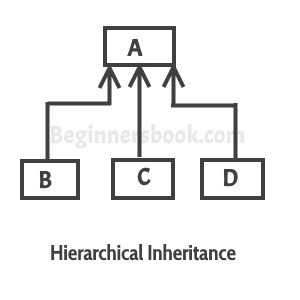


[**Multilevel inheritance**](https://beginnersbook.com/2013/12/multilevel-inheritance-in-java-with-example/)**:**

****class C extends class B and class B extends class A.

[**Hierarchical inheritance**](https://beginnersbook.com/2013/10/hierarchical-inheritance-java-program/)**:**

 classes B, C & D extends the same class A

****

[**Hybrid inheritance**](https://beginnersbook.com/2013/10/hybrid-inheritance-java-program/)**:**

Combination of more than one types of inheritance in a single program. For example, class A & B extends class C and another class D extends class A then this is a hybrid inheritance example because it is a combination of single and hierarchical inheritance.

***Java doesn’t support multiple inheritance where one call inherits multiple classes at the same time which is very complex which leads us to deadly diamond problem.***

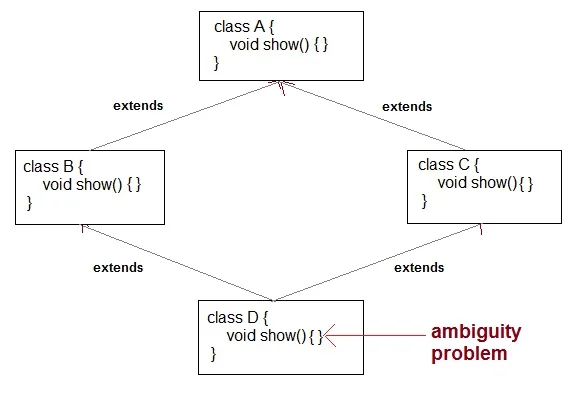
## **Disadvantages of Inheritance**

Main disadvantage of using inheritance is that the two classes (parent and child class) get tightly coupled.

This means that if we change code of parent class, it will affect to all the child classes which is inheriting/deriving the parent class, and hence, it cannot be independent of each other.

Java doesn’t support multiple inheritance because.

Deadly Dimond problem

******