Interface

**What is an interface in Java?**

Interface looks like a class but it is not a class. An interface can have methods and variables just like the class but the methods declared in interface are by default abstract (only method signature, no body)

**Why do we use interface ?**

* It is used to achieve total abstraction.
* Since java does not support multiple inheritance in case of class, but by using interface it can achieve multiple inheritance

interface MyInterface

{

/\* compiler will treat them as:

\* public abstract void method1();

\* public abstract void method2();

\*/

public void method1();

public void method2();

}

class Demo implements MyInterface

{

/\* This class must have to implement both the abstract methods

\* else you will get compilation error

\*/

public void method1()

{

System.out.println("implementation of method1");

}

public void method2()

{

System.out.println("implementation of method2");

}

public static void main(String arg[])

{

Demo obj = new Demo();

obj.method1();

}

}

## Interface and Inheritance

## Can we implement more than one interfaces in a class

Yes, we can implement more than one interfaces in our program because that doesn’t cause any ambiguity(see the explanation below).

interface X

{

public void myMethod();

}

interface Y

{

public void myMethod();

}

class JavaExample implements X, Y

{

public void myMethod()

{

System.out.println("Implementing more than one interfaces");

}

public static void main(String args[]){

JavaExample obj = new JavaExample();

obj.myMethod();

}

}

**Why Java doesn’t support Multiple Inheritance?**

// First Parent class

class Parent1

{

    void fun()

    {

        System.out.println("Parent1");

    }

}

// Second Parent Class

class Parent2

{

    void fun()

    {

        System.out.println("Parent2");

    }

}

// Error : Test is inheriting from multiple

// classes

class Test extends Parent1, Parent2

{

   public static void main(String args[])

   {

       Test t = new Test();

       t.fun();

   }

}

UML

**What is**[**UML**](https://www.geeksforgeeks.org/unified-modeling-language-uml-introduction/)**?**  
It is the general purpose modeling language used to visualize the system. It is a graphical language that is standard to the software industry for specifying, visualizing, constructing as well as for business modeling.

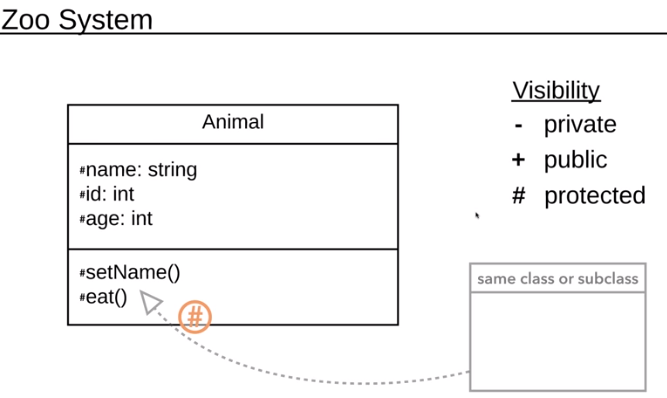
Why we need?

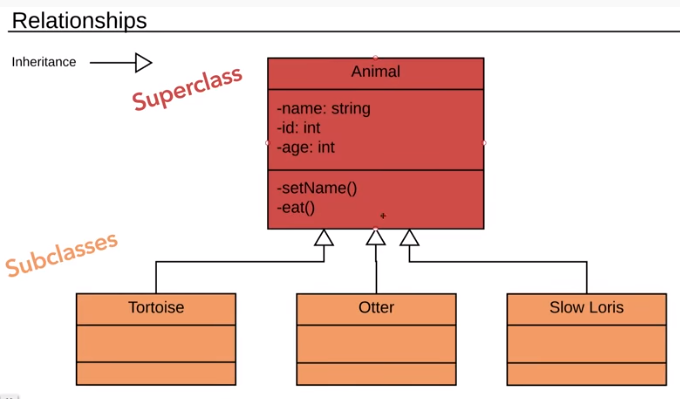
* It reduces thousands of words of explanation in a few graphical diagrams that may reduce time consumption to understand.
* It makes communication more clear and real.
* It helps to acquire the entire system in a view.
* It becomes very much easy for the software programmer to implement the actual demand once they have the clear picture of the problem.

**UML class diagrams:** Class diagrams are the main building blocks of every object oriented methods. The class diagram can be used to show the classes, relationships, interface, association, and collaboration. UML is standardized in class diagrams.

There are three types of modifiers which are used to decide the visibility of attributes and operations.

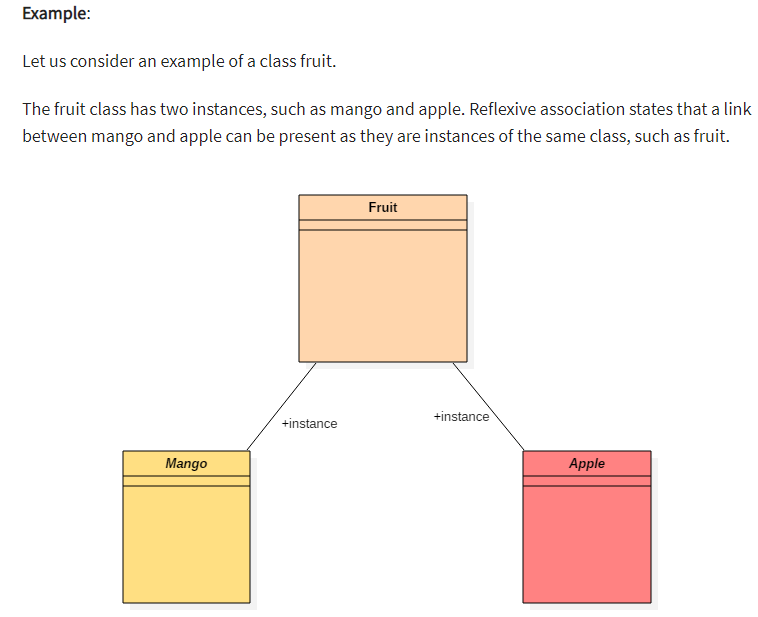
* + is used for public visibility(for everyone)
* # is used for protected visibility (for friend and derived)
* – is used for private visibility (for only me)

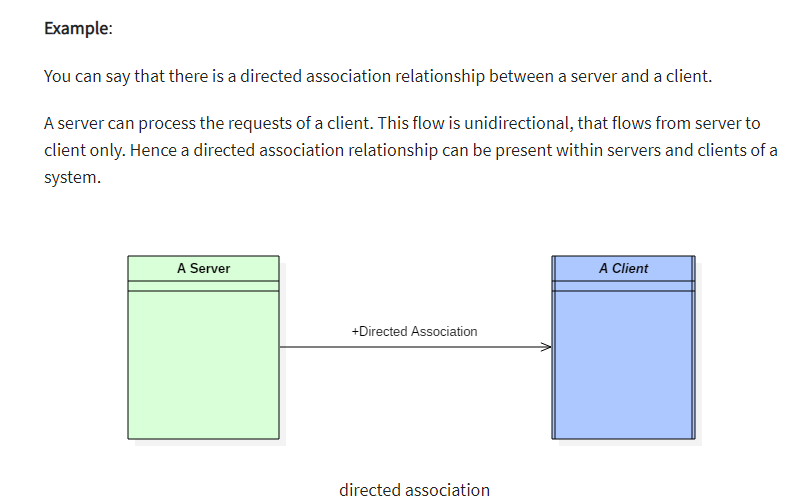
****

****

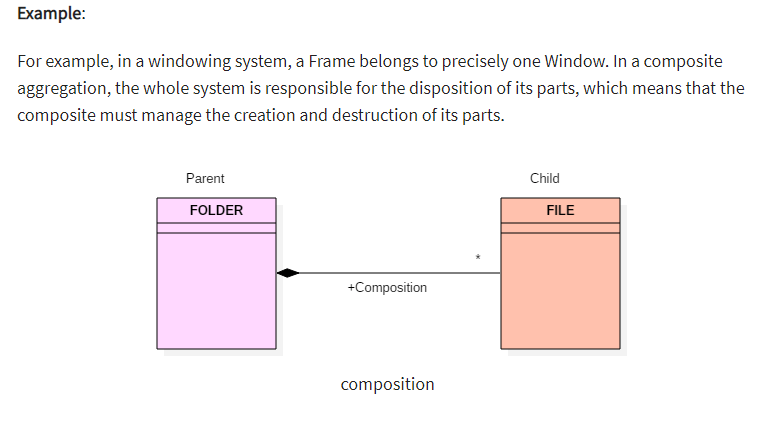
**Association**

It is a structural relationship that represents objects can be connected or associated with another object inside the system

****

****

**2. Composition** implies a relationship where the child cannot exist independent of the parent



The folder could contain many files, while each File has exactly one Folder parent. If a folder is deleted, all contained files are removed as well.

**3. In an aggregation** relationship, the dependent object remains in the scope of a relationship even when the source object is destroyed.

