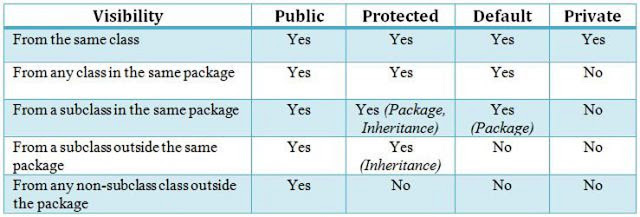
## Difference between public, protected and private modifier

Apart from the fact that each access modifier allows members a specific level of access, there is some more subtle difference between them e.g. where exactly can you use them. Let's find out more about different access modifiers in Java.  
  
1. One of the most important difference you need to remember related to access modifier is the level of accessibility e.g. anything public is accessible to anywhere and anything private is only accessible the class they are declared. Similarly, anything default is accessible only inside the package and anything protected is accessible outside the package but only to child classes.  
  
This means, you can access public class, method or variable from anywhere in Java program and that's why key API are always public e.g. List interface, ArrayList class etc but you might want to hide their implementation and you can do that by using private access modifier.  
  
Why do you want to hide implementation? so that you can change it later without any worry that it will break any client. One of the best examples of hiding implementation is RegularEnumSet and JumboEnumSet classes which implement EnumSet interface.  
  
EnumSet is public and these two classes are hidden at package level, which allows EnumSet to choose any of these depending upon key size of Enum. If the enum whose constants you are storing in EnumSet has less than 64 then it chooses RegularEnumSet which is more efficient.  
  
If you are looking for a Java development position and preparing for one then you can also take help from the wonderful book **Java Interview Exposed by Wrox**. It is one of the rare complete guides for a Java developer and tells which topic is important from interview point of view.

2. Second difference between public and modifier is that they provide Encapsulation, which is just opposite of accessibility. Encapsulation says hide anything which varies and that's why we hide implementations, public keyword provides lowest level of Encapsulation and private modifier provides highest level of Encapsulation in Java.  
  
  
3. Third difference between private and public modifier is that you can use public modifier with top level class but you cannot make a top level class private in Java. Though both public and private modifier can be used with class e.g. nested and inner classes. You can make an inner class private which means no one can access it outside enclosing class.  
  
Here is a nice table which compares access level provided by public, protected, private and default access modifier in Java.

  
  
As I said, use public modifier to define API and use private modifier to hide implementation, but you can also make constants public e.g. class variable which is also marked final and static.

## Difference between default and protected modifier in Java

So, now you know the difference between private and public modifier in Java and understand when to use public and private in Java. Now, its time to understand difference between default or package level access and protected modifier in Java.  
  
1) First **difference between default and protected access modifier in Java** is that default is nothing but package level accessibility i.e. if you don't provide any access modifier to a class, method or variable then Java by default make them accessible inside the package. If you cannot make a member private then your next best choice is package level access. That's why its also called package-private access. On the other hand, protected modifier provides more accessibility then default modifier. You can access a protected member outside the package, but only inside sub classes.  
  
For example if a method getVersion() is protected inside a class and they are inside build package and class is public then this method is accessible to all classes inside build package but only to sub classes outside build package.

package build;

public class Builder{

protected int getVersion(){

return 1;

}

}

and here is our sub class :

package gui;

public class Display extends *Builder* {

public void show(){

System.out.println(getVersion()); //Ok, protected method can be called from subclass outside package

}

}

You can see that we can access getVersion() method which is protected, inside another package gui without any problem.