Access modifiers in Java ?

** An access modifiers describes the **accessibility level of a class OR the member of the class (fields + methods)**

* In terms of accessibility, Java has provided 4 access modifiers :

a) private (Within the same class only)
b) default (Within the same package (folder) only)
c) protected (Within the same package as well as from another package but using Inheritance)
d) public (From everywhere)

Access Modifiers	Within the same class	Within the same Package	From Another Package	From everywhere (No restriction)
private	YES	NO	NO	NO
default	YES	YES	NO	NO
protected	YES	YES	YES [Using Inheritance]	NO
public	YES	YES	YES	YES

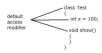
private access modifier :

It is the most restrictive access modifier because the member declared as private can't be accessible from outside of the class.

In Java we can't declare an outer class as a private or protected Or static. Generally we should declare the non static fields with private access modifier (Data hiding).

In java outer classes can be declared as public, abstract, final, sealed and non-sealed modifier only.

deafult access modifier :




```
class Test
{
int x = 100;
void show()
{
}
}
```

It is an access modifier which is less restrictive than private. It is such kind of access modifier whose physical existance is not available that means when we don't specify any kind of access modifier before the class name, field name or method name then by default it would be default.

As far as its accessibility is concerned, default members are accessible within the same folder(package) only. It is also known as private-package modifier.

```
package com.blc;

public class Test
{
    int x = 100;
}
```

```
package com.blc;

public class Main {

    public static void main(String[] args)
    {
        Test t1 = new Test();
        System.out.println(t1.x);

    }

}
```

Note : Main.java is available in the same package so, can access default member.

protected :

It is an access modifier which is less restrictive than default because the member declared as protected can be accessible from the outside of the package (folder) too but by using inheritance concept.

```
package com.blc;

public class Test
{
    protected int x = 500;
}
```

```
package com.elc;

import com.blc.Test;

public class ELC extends Test  //Inheritance
{
    public static void main(String[] args)
    {
        ELC e = new ELC();
        System.out.println(e.x);
    }
}
```

Note : ELC class is available in another package so It can access protected member by using Inheritance only.

public :

It is an access modifier which does not contain any kind of restriction that is the reason the member declared as public can be accessible from everywhere without any restriction.

According to Object Oriented rule we should declare the classes and methods as public where as fields must be declared as private or protected according to the requirement.

Note : If a method is used for internal purpose only (like validation) then we can declare that method as private method.It is called Helper method.

Access modifiers : private, default, protected and public

Remaining all are non access modifiers OR simply modifiers like final, static and so on

---

How to print Object properties (non static fields only) by using toString() method of Object class :

If we want to print our object properties (Instance Variables OR Non static fields) then we should generate(Override) toString() method in our class from Object class.

Now with the help of toString() method we need not to write any display kind of method to print the object properties i.e instance variable.

In order to generate the toString() method we need to follow the steps
Right click on the program -> source -> generate toString()

In order to call this toString() method, we need to print the corresponding object reference by using System.out.println() statement.

```
Manager m = new Manager();
System.out.println(m); //Calling toString() method of Manager class

Employee e = new Employee();
System.out.println(e); //Calling toString() method of Employee class.
```

```
package com.blc;

public class Manager
{
    public int managerId;
    private String managerName;
    private double managerSalary;

    public void setManagerData(int id, String name, double salary)
    {
        managerId = id;
        managerName = name;
        managerSalary = salary;
    }


    public String toString()
    {
      return "Manager [manager-Id=" + managerId + ", managerName=" + managerName + ",
managerSalary=" + managerSalary
              + "]";
    }

}
```

```
package com.elc;

import com.blc.Manager;

public class ManagerDemo
{
    public static void main(String[] args)
    {
        Manager smith = new Manager();
        smith.setManagerData(101, "Mr Smith", 85000);
        System.out.println(smith);

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

        Manager alex = new Manager();
        alex.setManagerData(102, "Mr Alex", 95000);
        System.out.println(alex);
    }

}
```