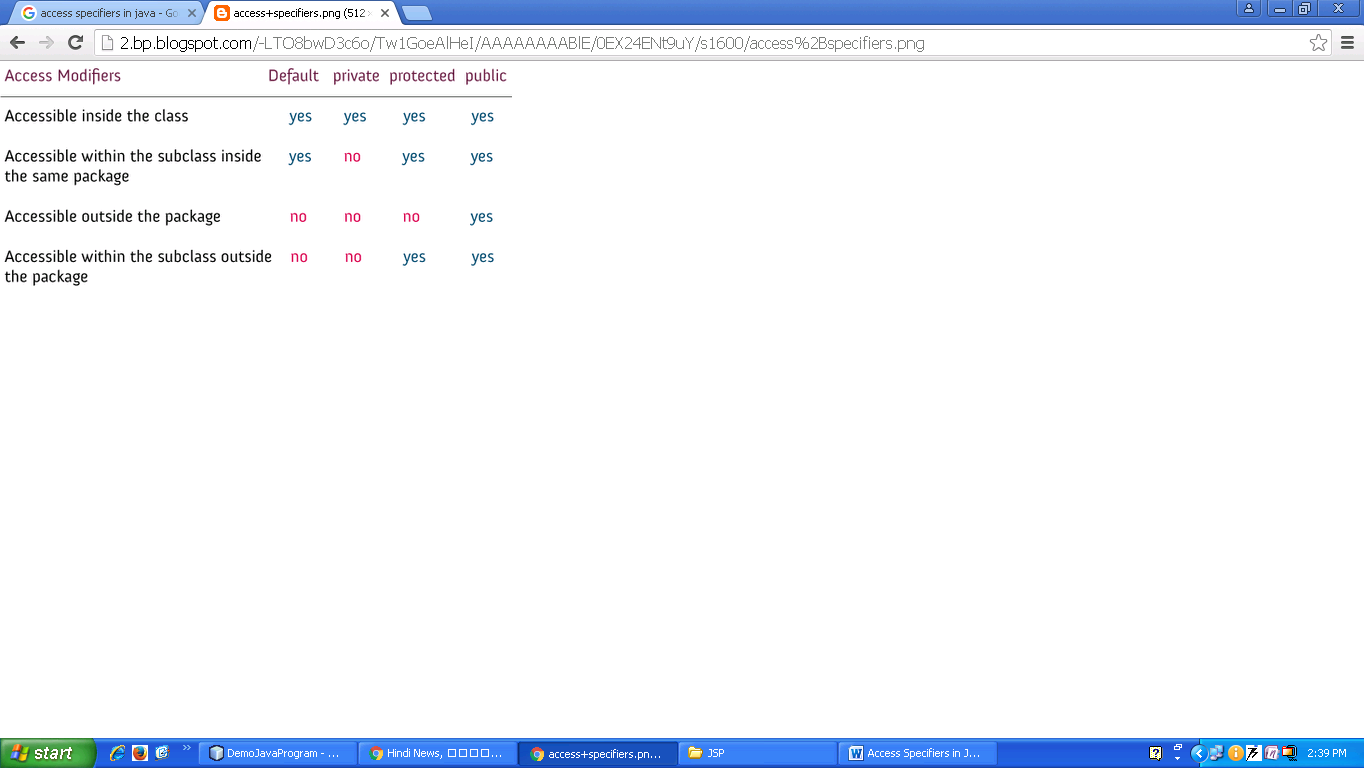
**Access Specifiers in Java**

* Java Access Specifiers (also known as Visibility Specifiers ) regulate access to classes, fields and methods in Java.
* Java Access Specifiers determine whether a field or method in a class, can be used or invoked by another method in another class or sub-class.
* Java Access Specifiers can be used to restrict access.
* Access Specifiers are an integral part of object-oriented programming.

Types Of Access Specifiers :  
  
In java we have four Access Specifiers and they are listed below.  
  
1.Public  
2.Private  
3.Protected  
4.Default(no specifier)



**Public specifiers :**

* Public specifiers achieve the highest level of accessibility.
* Classes, methods, and fields declared as public can be accessed from any class in the Java program, whether these classes are in the same package or in another package.

Eg.

public class Demo { // public class

public x, y, size; // public instance variables

public void find\_fact(){}// public method

}

**Private specifiers :**

* Private specifiers achieve the lowest level of accessibility.
* Private methods and fields can only be accessed within the same class to which the methods and fields belong.
* Private methods and fields are not visible within subclasses and are not inherited by subclasses. So, the private access specifier is opposite to the public access specifier.
* Using private specifier we can achieve encapsulation and hide data from the outside world.

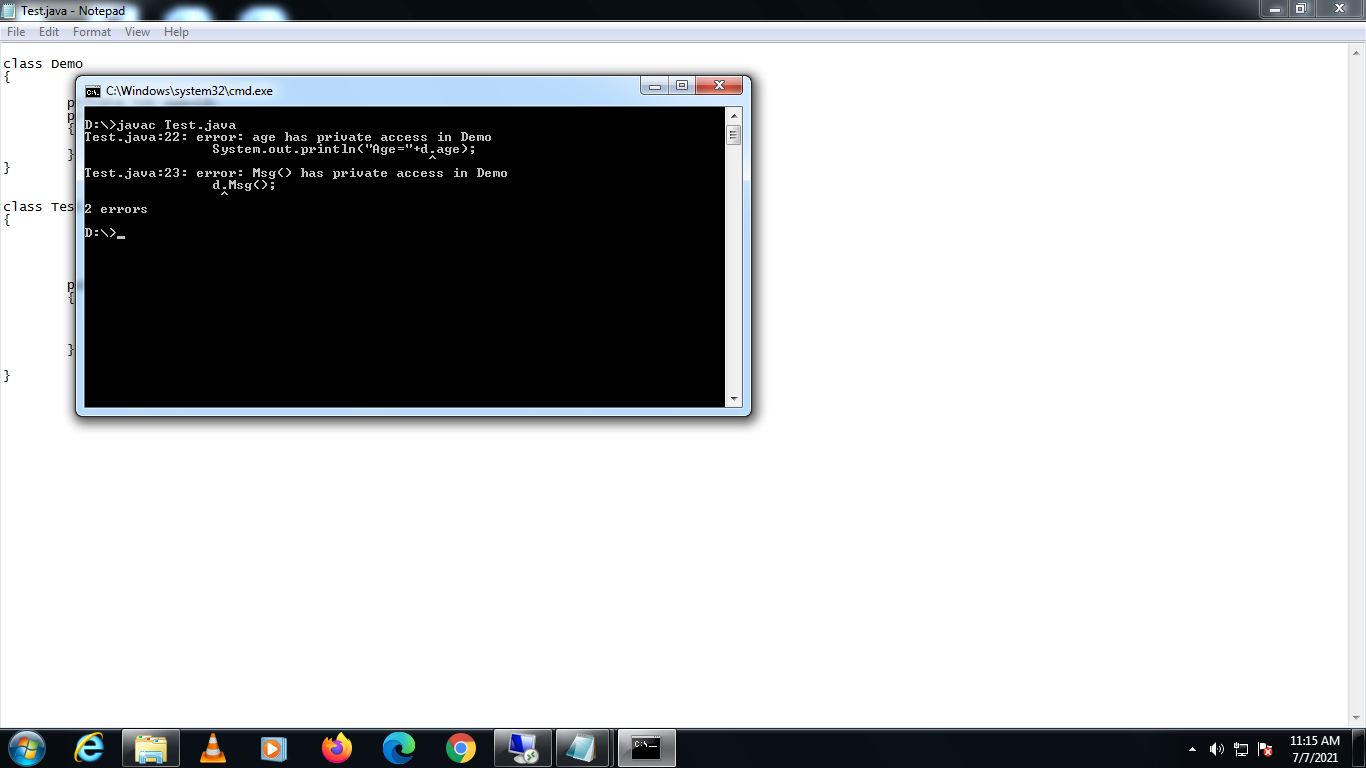
Eg

private double x, y; // private (encapsulated) instance variables

private void find\_fact(){}// private method

|  |
| --- |
| class Test  {    private int age=18;  private void Msg()  {  System.out.println("Age="+age);  }    public static void main(String args[])  {  Test d=new Test();  System.out.println("Age="+d.age);  d.Msg();  }  } |

|  |
| --- |
| class Demo  {  private int age=18;  private void Msg()  {  System.out.println("Age="+age);  }  }  class Test  {    public static void main(String args[])  {  Demo d=new Demo();  System.out.println("Age="+d.age);  d.Msg();  }  } |



**Protected specifiers :**

* Methods and fields declared as protected can be accessed by the subclasses in the same package or can be accessed by the subclasses in the other package.
* The protected access specifier cannot be applied to **class** and **interfaces**.

Eg

protected double x, y; // protected instance variables

protected void find\_fact(){} //protected method

**default(no specifier):**

* When you don't set access specifier for the element, it will follow the default accessibility level.
* There is no default specifier keyword.
* Classes, variables, and methods can be default accessed.
* Using default specifier we can access class, method, or field which belongs to same package, but not from outside this package.

Eg.

class Demo

{

int i; //Default access specifier

}

//Accessible within the subclass outside the package

package mypack;

import java.util.Scanner;

public class Person

{

int id;

String name;

protected void set\_data()

{

Scanner scan=new Scanner(System.in);

System.out.println("Enter the Id:");

id=scan.nextInt();

System.out.println("Enter the Name:");

name=scan.next();

}

protected void display()

{

System.out.print(id+"\t"+name+"\t");

}

};

import mypack.Person;

import java.util.Scanner;

class Test extends Person

{

intsal;

public void set\_data()

{

super.set\_data();

Scanner scan=new Scanner(System.in);

System.out.println("Enter the sal:");

sal=scan.nextInt();

}

public void display()

{

super.display();

System.out.print(sal);

}

public static void main(String args[])

{

Test p=new Test ();

p.set\_data();

p.display();

}

}