Java Access Specifiers (also known as Visibility Specifiers ) regulate access to classes, fields and methods in Java. These Specifiers determine whether a field or method in a class, can be used or invoked by another method in another class or sub-class. 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)  
  
We look at these Access Specifiers in more detail.  
  
[A picture containing calendar

Description automatically generated](https://2.bp.blogspot.com/-LTO8bwD3c6o/Tw1GoeAlHeI/AAAAAAAABlE/0EX24ENt9uY/s1600/access%2Bspecifiers.png)  
  
*public specifiers :*  
  
Public Specifiers achieves 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.  
  
**Example :**

public class Demo { // public class

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

}

*private specifiers :*  
  
Private Specifiers achieves 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.  
  
**Example :**

public class Demo { // public class

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

public set(int x, int y) { // setting values of private fields

this.x = x;

this.y = y;

}

public get() { // setting values of private fields

return Point(x, y);

}

}

*protected specifiers :*  
  
Methods and fields declared as protected can only be accessed by the subclasses in other package or any class within the package of the protected members' class. The protected access specifier cannot be applied to class and interfaces.  
  
*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.  
  
**Example :**

class Demo

{

int i; (Default)

}

Diagram

Description automatically generated