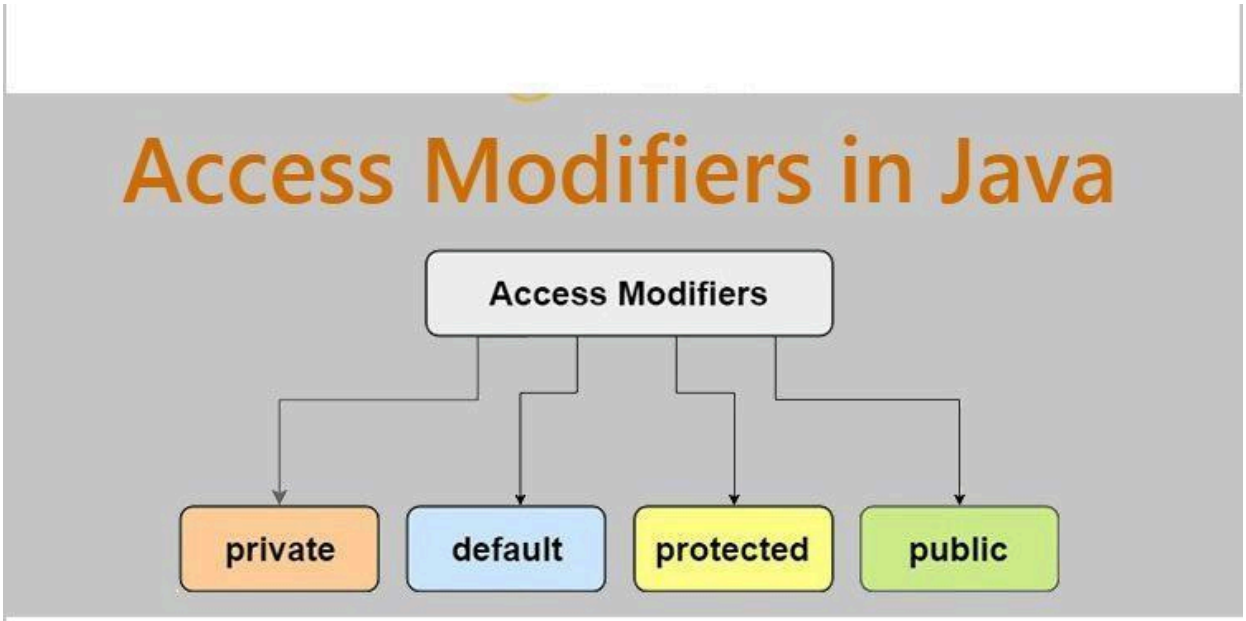
Access Modifiers in Java

In Java, Access modifiers helps to restrict the scope of a class, constructor, variable, method, or data member. It provides security, accessibility, etc. to the user depending upon the access modifier used with the element. In Java, there are four primary access modifiers:

1. Private
2. Default
3. Protected
4. Public



Each modifier defines a specific scope for accessing the class members. Let's dive into each of them along with examples.

Private: Private access modifier is the most restrictive modifier and allows classes, methods, and variables to be accessed only within the same class in which they are defined. Any class or method that is marked as private cannot be accessed from any other class or method in the program.

Example:

```
class A {
    private int x;

    int getX() {
        return x;
    }
}

class B {
    public static void main(String[] args) {
        A t = new A();
        // System.out.println(t.x); // Error: x has private access in A
        System.out.println(t.getX()); // Valid through getter method
    }
}
```

Explanation: The variable `x` is private, so it cannot be accessed directly in class `B`. However, the `getX()` method allows controlled access to `x`.

Default: Default access modifier is the one that is used when no access modifier is specified. It allows classes, methods, and variables to be accessed within the same package. Any class or method that is marked as default can be accessed from any other class or method within the same package.

Example:

```
package p1;
public class A {
    void fun() {
        System.out.println("Fun");
    }
}

package p2;
import p1.*;

class B {
```

```
    public static void main(String[] args) {
        A t = new A();
        // t.fun(); // Error: fun() has default access in p1.A
    }
}
```

Explanation: The method `fun()` is default, meaning it is accessible only within the `p1` package. Accessing it from package `p2` results in an error.

Protected: Protected access modifier allows classes, methods, and variables to be accessed within the same package or in a subclass outside of the package. Any class or method that is marked as protected can be accessed from any other class or method within the same package or in a subclass outside of the package.

Example:

```
package p1;
public class A {
    protected void fun() {
        System.out.println("Fun");
    }
}

package p2;
import p1.*;

class B extends A {
    public static void main(String[] args) {
        B t = new B();
        t.fun();
    }
}
```

Explanation: The method `fun()` is protected, so it is accessible in the subclass `B`, even though `B` is in a different package.

Public: Public access modifier is the least restrictive modifier and allows classes, methods, and variables to be accessed from anywhere in the program. Any class or

method that is marked as public can be accessed from any other class or method in the program.

Example:

```
package p1;
public class A {
    public void fun() {
        System.out.println("Fun");
    }
}

package p2;
import p1.*;

class B {
    public static void main(String[] args) {
        A t = new A();
        t.fun();
    }
}
```

Explanation: The method `fun()` is public, so it can be accessed from any package.

	default	private	protected	public
Same Class	Yes	Yes	Yes	Yes
Same package subclass	Yes	No	Yes	Yes
Same package non-subclass	Yes	No	Yes	Yes
Different package subclass	No	No	Yes	Yes
Different package non-subclass	No	No	No	Yes

Key Points to Remember

1. Encapsulation: Use the most restrictive access modifier necessary. For example, prefer `private` for data members unless sharing is explicitly needed.

2. Default Modifier: If no modifier is specified, it defaults to package-private visibility.
3. Inheritance with Protected: Protected members can be accessed in subclasses, even if they belong to a different package.
4. Public Classes and Members: Use public access only when the method or class is intended for universal access.
5. Nested Classes: Access modifiers can also be applied to nested classes.