Access Modifier

Access modifiers are keywords used in Java to define the accessibility or visibility of classes, methods, and variables. There are four types of access modifiers in Java:

Access Modifier	within class	within package	outside package by subclass only	outside package
Private	Y	N	N	N
Default	Y	Y	N	N
Protected	Y	Y	Y	N
Public	Y	Y	Y	Y

public:

The public access modifier allows unrestricted access to the class, method, or variable from any other class in any package. A public class, method, or variable can be accessed by any other class in the application.

protected:

The protected access modifier allows access to the class, method, or variable from within the same package and from any subclass, even if the subclass is in a different package. A protected class, method, or variable can be accessed by any subclass in the application.

default or package-private:

The default or package-private access modifier is used when no access modifier is specified explicitly. It allows access to the class, method, or variable from within the same package only. A default class, method, or variable can be accessed by any other class in the same package but not from outside the package.

private:

The private access modifier restricts access to the class, method, or variable to only within the same class. A private class, method, or variable cannot be accessed from any other class in the application.

BULLET POINTS

A class cannot be private or protected except nested class

The default modifier is more restrictive than protected

Protected methods can be accessed from outside the class only through inheritance

Modifier	Description
Default	declarations are visible only within the package (package private)
Private	declarations are visible within the class only
Protected	declarations are visible within the package or all subclasses
Public	declarations are visible everywhere

Let's see some examples of how to use these access modifiers in Java:

Access modifiers for class:

```
public class MyClass { // public class
    protected class MyInnerClass { // protected inner class
        // class definition
    }

    class AnotherInnerClass { // default or package-private inner class
        // class definition
    }

    private class MyPrivateInnerClass { // private inner class
        // class definition
    }
}
```

In this example, the MyClass class has three inner classes with different access modifiers: MyInnerClass is protected, AnotherInnerClass is default or package-private, and MyPrivateInnerClass is private.

Access modifiers for methods:

```java
public class MyClass {
    public void publicMethod() { // public method
        // method definition
    }

    protected void protectedMethod() { // protected method
        // method definition
    }

    void defaultMethod() { // default or package-private method
        // method definition
    }

    private void privateMethod() { // private method
        // method definition
    }
}
```

In this example, the MyClass class has four methods with different access modifiers: publicMethod is public, protectedMethod is protected, defaultMethod is default or package-private, and privateMethod is private.

Access modifiers for variables:

```java
public class MyClass {
    public int publicVar = 1; // public variable
    protected int protectedVar = 2; // protected variable
    int defaultVar = 3; // default or package-private variable
    private int privateVar = 4; // private variable
}
```

In this example, the MyClass class has four variables with different access modifiers: publicVar is public, protectedVar is protected, defaultVar is default or package-private, and privateVar is private.

In conclusion, access modifiers are an important aspect of Java that helps us to control the accessibility of classes, methods, and variables. It's important to choose the right access modifier based on the requirements and design of the application.