# Access Modifiers in Java

Access modifiers are keywords in Java that are used to set accessibility. An access modifier restricts the access of a class, constructor, data member and method in another class.

Java language has four access modifiers to control access level for classes and its members.

1. Default: The access level of a default modifier is only within the package. It cannot be accessed from outside the package. If you do not specify any access level, it will be the default.

2. Public: The access level of a public modifier is everywhere. It can be accessed from within the class, outside the class, within the package and outside the package.

3. Protected: The access level of a protected modifier is within the package and outside the package through child class. If you do not make the child class, it cannot be accessed from outside the package.

4. Private: The access level of a private modifier is only within the class. It cannot be accessed from outside the class.

| Accessibility / Access Specifier / Location | Same Class | Same Package | | Other Package | |
|---|---|---|---|---|---|
| | | Child class | Non-child class | Child class | Non-child class |
| Public | Yes | Yes | Yes | Yes | Yes |
| Protected | Yes | Yes | Yes | Yes | No |
| Default | Yes | Yes | Yes | No | No |
| Private | Yes | No | No | No | No |

1. Default Access Modifier

If we don’t specify any access modifier then it is treated as default modifier. It is used to set accessibility within the package. It means we can’t access it’s method or class from outside the package. It is also known as package accessibility modifier.

Example:

In this example, we created a Demo class and another class Test by which we are accessing show() method of Demo class. We did not mentioned access modifier for the show() method that’s why it is not accessible and reports an error during compile time.

```
class Demo
{
	int a = 10;
	// default access modifier
	void show() {
	System.out.println(a);
	}
}


class Test
{
	public static void main(String[] args)
	{
		Demo d = new Demo();
		d.show();
	}
}
```

2. Public Access Modifier

public access modifier is used to set public accessibility to a variable, method or a class. Any variable or method which is declared as public can be accessible from anywhere in the application.

Example:

Here, we have two class Demo and Test. Now we want to access show method of Demo class from Test class. The method has public accessibility so it works fine. See the below example.

Demo.java

```
package package1;

public class Demo {
  int a = 10;
  // public access modifier
  public void show() {
    System.out.println(a);
  }
}
```

Test.java

```
package package2;

import package1.Demo;

public class Test {
  public static void main(String[] args) {
    Demo demo = new Demo();
    demo.show();
  }
}
```

3. Protected Access Modifier

Protected modifier protects the variable, method from accessible from outside the class. It is accessible within class, and in the child class (inheritance) whether child is located in the same package or some other package.

Example:

In this example, Test class is extended by Demo and called a protected method show() which is accessible now due to inheritance.

Demo.java

```
package package1;

public class Demo {
  int a = 10;
  // public access modifier
  protected void show() {
    System.out.println(a);
  }
}
```

Test.java

```
package package2;

import package1.Demo;

public class Test extends Demo{
  public static void main(String[] args) {
    Test test = new Test();
    test.show();
  }
}
```

4. Private Access Modifier

Private modifier is most restricted modifier which allows accessibility within same class only. We can set this modifier to any variable, method or even constructor as well.

Example:

In this example, we set private modifier to show() method and try to access that method from outside of the class. Java does not allow to access it from outside of the class.

Demo.java

```
class Demo {
  int a = 10;
  private void show() {
    System.out.println(a);
  }
}
```

Test.java

```
public class Test {
  public static void main(String[] args) {
    Demo demo = new Demo();
    demo.show(); // compile error
  }
}
```