}

Access Modifiers in Java

Access modifiers in Java are keywords that determine the accessibility or scope of a class, constructor, method, or variable. They are used to implement encapsulation, one of the fundamental principles of object-oriented programming.

Types of Access Modifiers in Java

1. public
2. private
3. protected
4. default (no keyword)

1. public

Scope: Accessible from anywhere (same class, same package, different package).

Use Case: Use when the member should be accessible to all other classes.

Example:

```
public class Animal {
    public String name = "Lion";
    public void sound() {
        System.out.println("Roar");
    }
```

---

2. private

Scope: Accessible only within the same class.

Use Case: Use to hide implementation details and protect data.

Example:

```
public class Animal {
    private String name = "Tiger";
    private void sound() {
        System.out.println("Growl");
    }
```

```
   public void show() {
      System.out.println(name);
      sound();
   }
}
```

3. protected

Scope: Accessible within the same package and in subclasses of other packages.

Use Case: Use when inheritance is involved and you want to restrict general access.

Example:

```
package animals;
public class Animal {
   protected String name = "Elephant";
   protected void sound() {
      System.out.println("Trumpet");
   }
}
```

4. default (no modifier)

Scope: Accessible only within the same package.

Use Case: Use to restrict access to classes within the same package.

Example:

```
class Animal {
   String name = "Deer";
   void sound() {
      System.out.println("Bleat");
   }
}
```

Comparison Table

Access Modifier	**Same Class**	**Same Package**	**Subclass in Another Package**	**Other Packages**
public	Yes	Yes	Yes	Yes
protected	Yes	Yes	Yes	No
default	Yes	Yes	No	No
private	Yes	No	No	No

Conclusion

Access modifiers help in data hiding and security in Java applications. Choosing the appropriate modifier ensures proper encapsulation and controlled access to class members.