**Class Declaration and Modifiers**

A class may be a top-level class, nested class, derived class, super class, or an anonymous class—a class without a name. All these classes are described here.

1. Top-level class: It is a class that is not a nested class.

2. Nested class: It is a class that is declared within the body of another class or interface. There are four types of nested classes:

1. Nested static class: It is like any other static member of the enveloping class.

2. Nested non-static classes are also called inner classes and it can be classified as follows:

(a) Member inner class: It has properties like any other member of the outer class.

(b) Local class: A local class is defined within a method or code block.

(c) Anonymous class: As the name indicates, it is the inner class without a name.

3. Derived class or subclass: It is a class that is derived from (or extends) another class.

4. Super class: It is a class from which another class or other classes are derived.

5. Generic classes: These are classes that declare one or more type parameters.

6. Final class: It is a class that cannot be extended.

Access Modifiers - controls the access level

Non-Access Modifiers - do not control access level, but provides other functionality

**Non-Access Modifiers**

For **classes**, you can use either final or abstract:

Modifier

final: The class cannot be inherited by other classes (You will learn more about inheritance in the Inheritance chapter)

abstract: The class cannot be used to create objects (To access an abstract class, it must be inherited from another class. You will learn more about inheritance and abstraction in the Inheritance and Abstraction chapters)

For **attributes and methods**

|  |  |
| --- | --- |
| **Modifier** | **Description** |
| final | Attributes and methods cannot be overridden/modified |
| static | Attributes and methods belongs to the class, rather than an object |
| abstract | Can only be used in an abstract class, and can only be used on methods. The method does not have a body, for example **abstract void run();**. The body is provided by the subclass (inherited from). You will learn more about inheritance and abstraction in the [Inheritance](https://www.w3schools.com/java/java_inheritance.asp) and [Abstraction](https://www.w3schools.com/java/java_abstract.asp) chapters |
| transient | Attributes and methods are skipped when serializing the object containing them |
| synchronized | Methods can only be accessed by one thread at a time |
| volatile | The value of an attribute is not cached thread-locally, and is always read from the "main memory" |

**These are:**

**#1) Default:**Whenever a specific access level is not specified, then it is assumed to be ‘default’. The scope of the default level is within the package.

**#2) Public:**This is the most common access level and whenever the public access specifier is used with an entity, that particular entity is accessible throughout from within or outside the class, within or outside the package, etc.

**#3) Protected:**The protected access level has a scope that is within the package. A protected entity is also accessible outside the package through inherited class or child class.

**#4) Private:**When an entity is private, then this entity cannot be accessed outside the class. A private entity can only be accessible from within the class.

| **Access Specifier** | **Inside Class** | **Inside Package** | **Outside package subclass** | **Outside package** |
| --- | --- | --- | --- | --- |
| Private | Yes | No | No | No |
| Default | Yes | Yes | No | No |
| Protected | Yes | Yes | Yes | No |
| Public | Yes | Yes | Yes | Yes |