**OOPS**

**Java class**

 It is a user-defined blueprint or prototype from which objects are created.

Class does not occupy memory.

**Class Declaration in Java**

*access\_modifier* **class** <*class\_name*>

{

data member;

method;

constructor;

nested class;

interface;

}

In general, class declarations can include these components, in order:

1. ***Modifiers****: A class can be public or has default access (Refer*[*this*](https://www.geeksforgeeks.org/access-specifiers-for-classes-or-interfaces-in-java/)*for details).*
2. ***Class keyword:****class keyword is used to create a class.*
3. ***Class name:****The name should begin with an initial letter (capitalized by convention).*
4. ***Superclass(if any):****The name of the class’s parent (superclass), if any, preceded by the keyword extends. A class can only extend (subclass) one parent.*
5. ***Interfaces(if any):****A comma-separated list of interfaces implemented by the class, if any, preceded by the keyword implements. A class can implement more than one interface.*
6. ***Body:****The class body is surrounded by braces, { }.*

## Java Objects

Objects are the instances of a class that are created to use the attributes and methods of a class.

An object consists of :

1. **State**: It is represented by attributes of an object. It also reflects the properties of an object.
2. **Behavior**: It is represented by the methods of an object. It also reflects the response of an object with other objects.
3. **Identity**: It gives a unique name to an object and enables one object to interact with other objects.

*When we create an object which is a non primitive data type, it’s always allocated on the heap memory.*

When an object of a class is created, the class is said to be **instantiated**.

The new operator instantiates a class by allocating memory for a new object and returning a reference to that memory.

The new operator also invokes the class constructor.

## Ways to Create an Object of a Class

### ****Using new keyword****

### ****Using Class.forName(String className) method****

### ****3. Using clone() method****

In the inheritance system, we use a parent class reference variable to store a sub-class object. In this case, we can switch into different subclass objects using the same referenced variable.

**Example:**

class Animal {}

class Dog extends Animal {}

class Cat extends Animal {}

public class Test

{

// using Dog object

Animal obj = new Dog();

// using Cat object

obj = new Cat();

}

### Creating multiple objects by one type only (A good practice)

In real-time, we need different objects of a class in different methods. Creating a number of references for storing them is not a good practice and therefore we declare a static reference variable and use it whenever required. In this case, the wastage of memory is less. The objects that are not referenced anymore will be destroyed by the [Garbage Collector](https://www.geeksforgeeks.org/garbage-collection-java/) of Java.

In the inheritance system, we use a parent class reference variable to store a sub-class object. In this case, we can switch into different subclass objects using the same referenced variable.

**Example:**

class Animal {}

class Dog extends Animal {}

class Cat extends Animal {}

public class Test

{

// using Dog object

Animal obj = new Dog();

// using Cat object

obj = new Cat();

}

## Anonymous Objects in Java

Anonymous objects are objects that are instantiated but are not stored in a reference variable.

Java Methods

## Syntax of Method

<access\_modifier> <return\_type> <method\_name>( list\_of\_parameters)

{

//body

}

**Ways to Create Method in Java**

There are two ways to create a method in Java:

**1. Instance Method:** Access the instance data using the object name.Declared inside a class.

*Syntax:*

|  |
| --- |
| // Instance Method  **void** method\_name(){    body // instance area  } |

**2. Static Method:** Access the static data using class name. Declared inside class with **static**keyword.

*Syntax:*

|  |
| --- |
| //Static Method  **static** **void** method\_name(){    body // static area  } |

### Method Signature

It consists of the method name and a parameter list

# Access Modifiers in Java

### ****1. Default Access Modifier****

accessible **only within the same package**.

### ****2. Private Access Modifier****

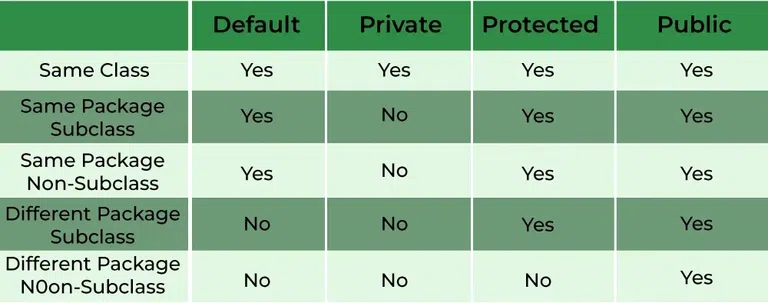
only **within the class** in which they are declared.

### ****3. Protected Access Modifier****

**accessible within the same package or subclasses in different packages.**

### ****Public Access modifier****

**accessible from everywhere**

****

STATIC KEYWORD

* **The static keyword:**When we declares a class as static then it can be used without the use of an object in Java. If we are using static function or static variable then we can’t call that function or variable by using dot(.) or class object defying object oriented feature.