

Q) What do you mean by static keyword? What are static variables or class variables in Java?

## STATIC Keyword

→ Data members

→ Benefit: →  
memory efficiency

Property not associated with any object

Instance/object variable (nonStatic)

vs class variable (static)

Accessing static variables

using classname (without object)

eg Company name of Employees, college name of Students  
language in Bollywood Movie Industry, etc.

```

class Movie {
    // Non Static or Instance Variables
    int duration;
    String name;
    double rating;

    // Static or Class Variable
    static String language = "English";

    public Movie(String name, int duration, double rating) {
        this.name = name;
        this.duration = duration;
        this.rating = rating;
    }
}

```

```

public static void main(String[] args) {
    Movie a1 = new Movie(name: "Avengers Infinity War", duration: 130, rating: 4.8);
    Movie a2 = new Movie(name: "Avengers End Game", duration: 200, rating: 4.9);
    Movie a3 = new Movie(name: "Avengers Secret Wars", duration: 170, rating: 4.6);
    Movie a4 = new Movie(name: "Avengers Kang Dynasty", duration: 220, rating: 5.0);

    // Non Static Variables
    System.out.println(a1.name + " " + a1.duration + " " + a1.rating);
    System.out.println(a2.name + " " + a2.duration + " " + a2.rating);
    System.out.println(a3.name + " " + a3.duration + " " + a3.rating);
    System.out.println(a4.name + " " + a4.duration + " " + a4.rating);

    // Static Variable or Class Variable
    System.out.print(a1.language + " ");
    System.out.print(a2.language + " ");
    System.out.print(a3.language + " ");
    System.out.print(a4.language + " ");

    // Accessing Using Class Name
    // (Static Properties Even Exist Without Single Object)
    System.out.print(Movie.language + " ");
}

```

```

● architagarwal@Archits-MacBook-Air 00PS_Codes % javac 11.StaticKeyword.java
● architagarwal@Archits-MacBook-Air 00PS_Codes % java Driver
Avengers Infinity War 130 4.8
Avengers End Game 200 4.9
Avengers Secret Wars 170 4.6
Avengers Kang Dynasty 220 5.0
English English English English English %

```

Interview  
Ques

Q) Why are static variables considered evil?

- It represents global scope (accessible using classname & from all objects of class)
- lifetime of variable is very long (entire program)
- less Object-Oriented way and reduce reusability
  - class loading init
  - donot require object creation
  - not object specific
- Not thread safe and difficult to serialize  
(multiple threads will see same instance/copy of static variables which can lead to inconsistency/race condition during concurrent/parallel access).







Q) What are static member functions in java? What are rules for accessing static or non-static data?

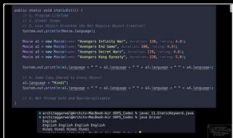
Static  
Member functions

Static methods  
can't access non-static  
methods & data members.

↳ Although we can still  
\* access by creating objects within static method.

Static method can access other static  
methods and static data members

Non-static methods can access both  
non-static as well as static members  
(data members & other methods)



## Movie Class

```
public void nonStaticFun() {
    // Access Non Static Members (Properties & Other Functions) : Allowed
    System.out.println(" Name : " + this.name + " Duration : "
        + this.duration + " Rating : " + this.rating);

    // Access Static Member (Properties & Other Functions) : Allowed
    System.out.println("Language : " + Movie.language);
    Movie.staticFun();
}

public static void staticFun() {
    // This Keyword is Not There Inside Static Functions
    // System.out.println(this);

    // Access Non Static Members (Properties & Other Functions) : Not Allowed
    // System.out.println(" Name : " + this.name + " Duration : " + this.duration +
    // " Rating : " + this.rating);
    // nonStaticFun();

    // Access Static Member (Properties & Other Functions) : Allowed
    System.out.println("Language : " + Movie.language);

    // Access Non Static Members Using Object Creation
    Movie obj = new Movie(name: "Avengers", duration: 150, rating: 4.5);
    System.out.println(obj.name + " " + obj.duration + " " + obj.rating);
}
```

## Output

```
architaggarwal@Archits-MacBook-Air OOPS_Codes % javac 11.StaticKeyword.java
architaggarwal@Archits-MacBook-Air OOPS_Codes % java Driver
Language : English
Avengers 150 4.5
Name : Avengers Infinity War Duration : 130 Rating : 4.8
Language : English
Language : English
Avengers 150 4.5
```

## Driver Code

```
// No Object Required for Calling Static Functions
Movie.staticFun();

Movie a1 = new Movie(name: "Avengers Infinity War", duration: 130, rating: 4.8);
// Object Required to Call Non Static Member Function
a1.nonStaticFun();
```

## Interview ques

→ entry point

Q) Why is `main()` method static in Java?

- JVM can call it without object creation
- It is the first method to undergo execution
- JVM can call `main()` method directly using classname

Note :- It should also be always →

- ① public
- ② void return type
- ③ `String[] args`

```
"public static void main(String[] args)"
```



### Interview Question

Q) Do we have "this" inside a "static" method?

A) No, since static methods are not associated with objects and this keyword is associated with a particular instance/object, hence we do not have "this" access inside "static" method.

### Interview Question

Q) Can constructors be static?

A) No, since constructors implicitly return the current object reference ("this"), they cannot be static.

Interview Question  
Q) Why is method static in Java?  
A) 1) static methods are associated with the class, not with any object.  
2) static methods can be called without creating an object of the class.  
3) static methods are shared by all objects of the class.  
4) static methods are used to perform operations on the class as a whole.  
5) static methods are used to perform operations on the class variables.  
6) static methods are used to perform operations on the class constants.  
7) static methods are used to perform operations on the class arrays.  
8) static methods are used to perform operations on the class collections.  
9) static methods are used to perform operations on the class streams.  
10) static methods are used to perform operations on the class sockets.  
11) static methods are used to perform operations on the class threads.  
12) static methods are used to perform operations on the class timers.  
13) static methods are used to perform operations on the class utilities.  
14) static methods are used to perform operations on the class validators.  
15) static methods are used to perform operations on the class wrappers.  
16) static methods are used to perform operations on the class writers.  
17) static methods are used to perform operations on the class zip files.  
18) static methods are used to perform operations on the class zip files.  
19) static methods are used to perform operations on the class zip files.  
20) static methods are used to perform operations on the class zip files.



Q) What do you understand by static nested classes in java?

Static  
Nested class → To create objects (instantiate)  
inner class objects without instantiating  
outer class

- Outer class can access inner class data members
  - Inner class can only access outer class' static members
  - Inner class object creation requires:- Outer class Name, inner class Name
- Note: If main() method & inner class are in the same outer class,  
we can omit outer class name

inner class can be static or non-static  
1) If inner class is static, it is associated with the class and not with the object.  
2) If inner class is non-static, it is associated with the object and not with the class.  
3) Inner class can be final.  
4) Inner class can be abstract.  
5) Inner class can be synchronized.  
6) Inner class can be volatile.  
7) Inner class can be transient.  
8) Inner class can be native.  
9) Inner class can be synchronized.  
10) Inner class can be volatile.  
11) Inner class can be transient.  
12) Inner class can be native.

```

class Movie {
    // Non Static or Instance Variables (Outer Class)
    int duration;
    String name;
    double rating;

    // Static or Class Variable (Outer Class)
    static String language = "English";

    public Movie(String name, int duration, double rating) {
        this.name = name;
        this.duration = duration;
        this.rating = rating;
    }

    public void outerNonStaticFun() { ...
    public static void outerStaticFun() { ...
    public static class Theater { ...
}

```

Outer class  
(Movie)

```

public void outerNonStaticFun() {
    // Access Outer Non Static Members (Properties & Other Functions) : Allowed
    System.out.println(" Name : " + this.name + " Duration : "
        + this.duration + " Rating : " + this.rating);

    // Access Outer Static Member (Properties & Other Functions) : Allowed
    System.out.println("Language : " + Movie.language);

    // Access Static Inner Class Static Members : Allowed
    System.out.println(Theater.screenType);

    // Access Static Inner Class Non Static Members
    // Without Inner Class Object: Not Allowed
    // System.out.println(this.chain + " " + this.noOfSeats);

    // With Inner Class Object: Allowed
    Theater inner = new Theater(chain: "PVR", noOfSeats: 100);
    System.out.println(inner.chain + " " + inner.noOfSeats);

    System.out.println(x: "-----");
}

```

```

public static void outerStaticFun() {
    // Access Outer Non Static Members (Properties & Other Functions) : Not Allowed
    // System.out.println(" Name : " + this.name + " Duration : " + this.duration +
    // " Rating : " + this.rating);
    // outerNonStaticFun();

    // Access Outer Static Member (Properties & Other Functions) : Allowed
    System.out.println("Language : " + Movie.language);

    // Access Static Inner Class Static Members : Allowed
    System.out.println(Theater.screenType);

    // Access Static Inner Class Non Static Members
    // Without Inner Class Object: Not Allowed
    // System.out.println(this.chain + " " + this.noOfSeats);

    // With Inner Class Object: Allowed
    Theater inner = new Theater(chain: "PVR", noOfSeats: 100);
    System.out.println(inner.chain + " " + inner.noOfSeats);

    System.out.println(x: "-----");
}

```

# Inner class (Theater)

```
public static class Theater {
    // Non Static or Instance Variables (Inner Class)
    String chain;
    int noOfSeats;

    // Static or Class Variable (Inner Class)
    static String screenType = "2D";

    public Theater(String chain, int noOfSeats) {
        this.chain = chain;
        this.noOfSeats = noOfSeats;
    }

    public void innerNonStaticFun() {
        // Access Outer Non Static Members : Not Allowed
        // System.out.println(" Name : " + this.name + " Duration : "
        // + this.duration + " Rating : " + this.rating);

        // Access Outer Static Member (Properties & Other Functions) : Allowed
        System.out.println("Language : " + Movie.language);

        // Access Static Inner Class Static Members : Allowed
        System.out.println(Theater.screenType);

        // Access Static Inner Class Non Static Members: Allowed
        System.out.println(this.chain + " " + this.noOfSeats);

        System.out.println(x: "-----");
    }
}
```

```
public static void innerStaticFun() {
    // Access Outer Non Static Members (Properties & Other Functions) : Not Allowed
    // System.out.println(" Name : " + this.name + " Duration : " + this.duration +
    // " Rating : " + this.rating);
    // outerNonStaticFun();

    // Access Outer Static Member (Properties & Other Functions) : Allowed
    System.out.println("Language : " + Movie.language);

    // Access Static Inner Class Static Members : Allowed
    System.out.println(Theater.screenType);

    // Access Static Inner Class Non Static Members
    // Without Inner Class Object: Not Allowed
    // System.out.println(this.chain + " " + this.noOfSeats);

    // With Inner Class Object: Allowed
    Theater inner = new Theater(chain: "PVR", noOfSeats: 100);
    System.out.println(inner.chain + " " + inner.noOfSeats);

    System.out.println(x: "-----");
}
```



## Driver code

```
class Driver {  
    Run | Debug  
    public static void main(String[] args) {  
        Movie.outerStaticFun();  
        Movie.Theater.innerStaticFun();  
  
        Movie outer = new Movie(name: "Avengers", duration: 180, rating: 4.5);  
        outer.outerNonStaticFun();  
  
        Movie.Theater inner = new Movie.Theater(chain: "PVR", noOfSeats: 100);  
        inner.innerNonStaticFun();  
    }  
}
```

## Output

Language : English  
2D  
PVR 100  
-----

Language : English  
2D  
PVR 100  
-----

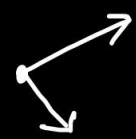
Name : Avengers Duration : 180 Rating : 4.5  
Language : English  
2D  
PVR 100  
-----

Language : English  
2D  
PVR 100  
-----



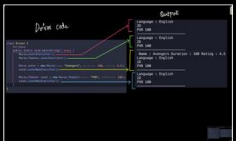


Q) What do you understand by "static block" in Java?

→ Static block  used to initialize static variables  
executed before main() execution  
at time of class loading

Q) Can you run a java program only using static block, i.e. without main function?

No, in newer Java versions, main() is mandatory to run a java program. If there is no entry point, java code will not run even if there is a static block.



```

class Movie {
    // Non Static or Instance Variables
    int duration;
    String name;
    double rating;

    // Static or Class Variable
    static String language;

    public Movie(String name, int duration, double rating) {
        System.out.println(x: "Constructor - Object Creation");
        this.name = name;
        this.duration = duration;
        this.rating = rating;
        this.language = "Hindi";
    }

    // Static Block (Runs Before Main Method Execution: During Loading & Linking)
    static {
        System.out.println(x: "Movie Static Block Executed");

        // Non Static Member Initialized
        language = "English";

        System.out.println(language + " OR " + Movie.language);
    }
}

```

```

class Driver {
    static int x;
    static {
        System.out.println(x: "Driver Static block executed");
        x = 100;
        System.out.println(x);
    }

    Run | Debug
    public static void main(String[] args) {
        System.out.println(x: "Main Method Started");

        // main method must be there otherwise static block will also not run,
        // Although it will run after static block of it's class

        System.out.println(x);
        System.out.println(Movie.language);

        Movie obj = new Movie(name: "Avengers", duration: 180, rating: 4.5);
        System.out.println(obj.language);
    }
}

```

```

● architagarwal@Archits-MacBook-Air OOPS Codes % javac 11.3.StaticBlock.java
● architagarwal@Archits-MacBook-Air OOPS Codes % java Driver
Driver Static block executed
100
Main Method Started
100
Movie Static Block Executed
English OR English
English
Constructor - Object Creation
Hindi

```

1) Static block is executed by JVM before the start of the main method.  
 → Static block is used to initialize static variables.  
 → It is executed before the start of the main method.  
 2) It is used to perform some operations on the class level.  
 → It is used to initialize static variables.  
 → It is used to perform some operations on the class level.  
 → It is used to initialize static variables.  
 → It is used to perform some operations on the class level.