**Static Variable in Java**

**Static variable in Java** is variable which belongs to the class and initialized only once at the start of the execution. It is a variable which belongs to the class and not to object(instance ). Static variables are initialized only once, at the start of the execution. These variables will be initialized first, before the initialization of any instance variables.

* A single copy to be shared by all instances of the class
* A static variable can be accessed directly by the class name and doesn’t need any object
  + **Syntax: <class\_name>.<static\_variable\_name>;**

## Static Method

When a method is declared with the static keyword, it is known as the static method. Static method in Java is a method which belongs to the class and not to the object. A static method can access only static data.

* A static method can call only other static methods and can not call a non-static method from it.
* A static method can be accessed directly by the class name and doesn’t need any object
* A static method cannot refer to “this” or “super” keywords in anyway

## ****Static Block****

The static block is a block of statement inside a Java class that will be executed when a class is first loaded into the JVM. A static block helps to initialize the static data members, just like constructors help to initialize instance members.

class Test{

static {

//Code goes here

}

}

**Final Keyword in Java**

The final modifier applies to classes, methods, and variables. The meaning of final varies from context to context, but the essential idea is the same.

* A final class cannot be inherited
* A final variable becomes a constant and its value cannot be changed.
* A final method cannot be overridden. This is done for security reasons, and these methods are used for optimization.

**Final Variables**

When a variable is declared with final keyword, its value can’t be modified, essentially, a constant.

Syntax:

* **a final static variable PI**

**static final double PI = 3.141592653589793;**

**Initializing a final Variable**

**We must initialize a final variable, otherwise, the compiler will throw a compile-time error.**

**There are three ways to initialize a final variable:**

1. **You can initialize a final variable when it is declared. This approach is the most common. A final variable is called a blank final variable if it is not initialized while declaration. Below are the two ways to initialize a blank final variable.**
2. **A blank final variable can be initialized inside the constructor. If you have more than one constructor in your class then it must be initialized in all of them, otherwise, a compile-time error will be thrown.**
3. **A blank final static variable can be initialized inside a static block.**

## Abstraction in Java

**Abstraction in JAVA** shows only the essential attributes and hides unnecessary details of the object from the user. In Java, abstraction is accomplished using Abstract class, Abstract methods, and Interfaces. Abstraction helps in reducing programming complexity and effort.

## Abstract Class

A class which is declared as abstract is known as an **abstract class**. It can have abstract and non-abstract methods. It needs to be extended and its method implemented. It cannot be instantiated.

* An abstract class must be declared with an abstract keyword.
* It can have abstract and non-abstract methods.
* It cannot be instantiated. We cannot create object of the abstract class
* It can have [constructors](https://www.javatpoint.com/java-constructor) and static methods also.
* It can have final methods which will force the subclass not to change the body of the method.
* For design purpose, a class can be declared abstract even if it does not contain any abstract methods
* Reference of an abstract class can point to objects of its sub-classes thereby achieving run-time polymorphism Ex: Shape obj = new Rectangle();

## Abstract Method

A method which is declared as abstract and does not have implementation is known as an abstract method.

**Example of abstract method:**

**abstract class Bike {**

**abstract void run();**

**}**

**Rules of Abstract Method**

* Abstract methods do not have an implementation. it only has method signature
* If a class is using an abstract method, they must be declared abstract. The opposite cannot be true. This means that an abstract class does not necessarily have an abstract method.
* If a regular class extends an abstract class, then that class must implement all the abstract methods of the abstract parent