# Reading Material

Introduction to Java

Haaris Infotech

SM- Week1 and 2-Part1

# INTRODUCTION TO JAVA

This section introduces you with JAVA. Java is an Object-Oriented Programming similar to C++ and Smalltalk. But JAVA is platform independent and follows the principle of WORA (Write Once and Run Anywhere). It is unlike C, C++ which means that you write a JAVA program and compile it only once and run it on any operation system be it Windows, Linux, Solaris etc.

If we talk about C++ then it is not all platform independent because needs to be compiled on each operation system in order to execute the program on different operation system. We will discuss this in detail in section Java Virtual Machine that how it achieves such a platform independence whereas C, C++, FORTRAN, VB, VC++ and many others.

## HISTORY OF JAVA

Java was conceived by James Gosling, Patrick Naughton, Chris Warth, Ed Frank, and Mike Sheridan at Sun Microsystems, Inc. in 1991. It took 18 months to develop the first working version. This language was initially called "Oak" but was renamed to "Java" in 1995 due to a patent search which determined that the name was copyrighted and used for another programming language. Between the initial implementation of Oak in the fall of 1992 and the public announcement of Java in the spring of 1995, many more people contributed to the design and evolution of the language. Bill Joy, Arthur van Hoff, Jonathan Payne, Frank Yellin, and Tim Lindholm were key contributors to the maturing of the original prototype.

Java has travelled a long way and it has made a mark of presence in 21st Century. Today, Java is used for many types of applications like embedded applications, financial applications, desktop applications, flight simulators, Image Processors, Games, distribute enterprise applications called J2EE and many more.

## WHY TO LEARN JAVA?

- Java works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc.)
- It is one of the most popular programming languages in the world
- It is easy to learn and simple to use
- It is open-source and free
- It is secure, fast and powerful
- It has a huge community support (tens of millions of developers)
- Java is an object-oriented language which gives a clear structure to programs and allows code to be reused, lowering development costs
- As Java is close to C++ and C#, it makes it easy for programmers to switch to Java or vice versa

### Java is easy to learn:

Java is quite easy to learn and can be understood in a short span of time as it has a syntax similar to English.

### Java has a large community:

There is a large online community of Java users ranging from beginner, advanced and even expert levels that are particularly helpful in case any support is required.

### Java has an abundant API:

Java has an abundant Application Programming Interface (API) that includes many Java classes, packages, interfaces, etc. This is useful for constructing applications without necessarily knowing their inside implementations.

Java has mainly three types of API i.e., **Official Java core API's, Optional official Java API's and Unofficial API's.** These APIs overall are used for almost everything including networking, I/O, databases, media, XML parsing, speech synthesis, etc.

### Java has multiple Open-Source Libraries:

Open-source libraries have resources that can be copied, studied, changed, shared, etc. There are multiple open-source libraries in Java such as JHipster, Maven, Google Guava, Apache Commons, etc. that can be used to make Java development easier, cheaper and faster.

### Java has Powerful Development Tools:

There are many Integrated development environments (IDE's) in Java that provides various facilities for software development to programmers. Powerful Java IDE's such as Eclipse, NetBeans, IntelliJ IDEA, etc. play a big role in the success of Java. These IDEs provide many facilities such as debugging, syntax highlighting, code completion, language support, automated refactoring, etc. that make coding in Java easier and faster.

Java has created a base for the Android operating system and opted around 90% fortune 500 companies for develop a lot of back-end applications. Also, it plays a great role in Apache Hadoop data processing, Amazon Web Services, and Windows Azure, etc.

### Java is Free of Cost:

One of the reasons Java is very popular among individual programmers is that it is available under the Oracle Binary Code License (BCL) free of charge. This means that Java is free for development and test environments, but for commercial purposes, a small fee is required.

### Java is Platform Independent:

Java is platform-independent as the Java source code is converted to byte code by the compiler which can then be executed on any platform using the Java Virtual Machine. Java is also known as a WORA (write once, run anywhere) language because it is platform-independent.

Also, the development of most Java applications occurs in a Windows environment while they are run on a UNIX platform because of the platform-independent nature of Java.

## Java has great Documentation Support:

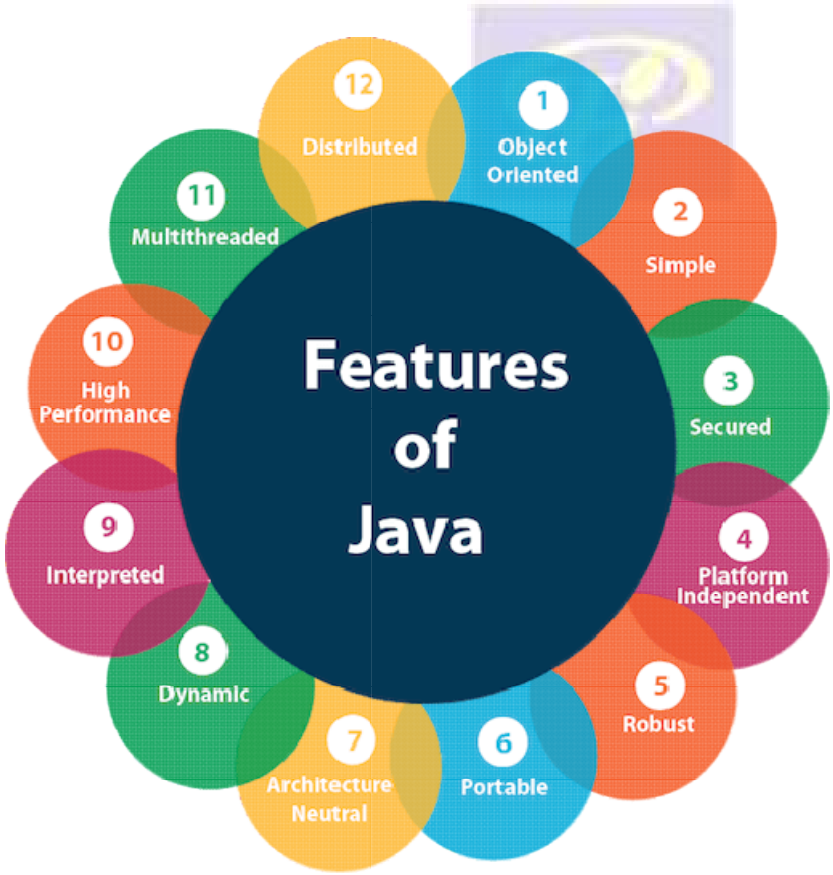
The documentation support for Java is excellent using Javadoc which is the documentation generator for Java. It uses the Java source code to generate the API documentation in HTML format. So, Javadoc provides a great reference while coding in Java so that understanding the code is quite simple.

## Java is Versatile:

Java is very versatile as it is used for programming applications on the web, mobile, desktop, etc. using different platforms. Also, Java has many features such as dynamic coding, multiple security features, platform-independent characteristics, network-centric designing, etc. that make it quite versatile.

Features of Java

A list of most important features of Java language is given below.




1. Simple

2. Object-Oriented
3. Portable
4. Platform independent
5. Secured
6. Robust
7. Architecture neutral
8. Interpreted
9. High Performance
10. Multithreaded
11. Distributed
12. Dynamic

## JDK, JRE, JVM

### Java Virtual Machine (JVM)

Java Virtual Machine is a small application mainly written in C language that needs to be installed in order to execute JAVA programs. Whenever a Java program is compiled, it is converted into BYTE CODE. Further when java program is executed, JVM executes those byte codes and converts them into machine language understood by the underlying operating system. Since JVM is platform dependent it makes Java platform independent. Byte codes produced during compilation are always same for all operating systems but since JVM is platform dependent it reads those byte codes and convert them into the machine language for which the JVM has been designed. e.g. JVM built for Linux operating system will convert the byte codes into machine language understood by Linux and the underlying machine hardware.

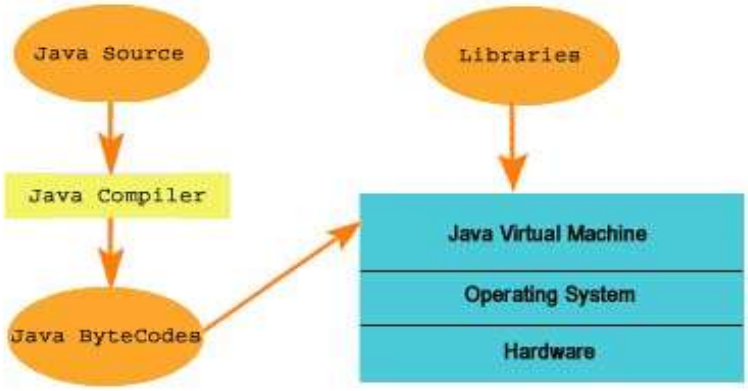
Most modern languages are designed to be compiled, not interpreted, mostly out of performance concerns. However, the fact that a Java program is executed by

the JVM helps solve the major problems associated with downloading programs over the Internet.

If a Java program were compiled to native code, then different versions of the same program would have to exist for each type of CPU and operating system connected to the Internet, which is not a feasible solution. Thus, the implementation of bytecode is the easiest way to create truly portable programs. Interpreted Java code also helps to make it secure. Since the execution of a Java program is under the control of the JVM, the JVM can control the program, and prevent it from generating side effects outside of it. This makes it ideal for applets to be downloaded from across the Internet, and executed on any CPU and Operating System that has the corresponding JVM implementation.

Java Development Kit (JDK) is a software development tool used for developing Java programs. It works as a compiler and a debugging tool to develop applets and applications. JDK is a set of development tools such as JavaDoc, Java Debugger, etc., and JRE to execute the program. It is platform dependent and used by Java developers. JDK can be used in Windows, MacOS systems.

Java Runtime Environment (JRE) is a software tool that provides tools which are necessary for the software to execute Java applications. JRE consists of class libraries, JVM, loader class. JRE comes along with JDK tools and need not to install JRE separately. JRE is used by those who want to run the Java Programs i.e., end users of your system. It is important to use JRE if we have to run Java applets.




## Java in Time Compiler (JIT)

Although Java was designed for interpretation, there is technically nothing about Java that prevents on-the-fly compilation of bytecode into native code. Sun provides its Just in Time (JIT) compiler for bytecode which when included in the JVM, it compiles bytecode into executable code in real time one by one as the program demands. The JIT compiles code as it is needed during execution. However, the just-in-time approach still yields a significant performance boost. Whether your Java program is actually taken in the traditional way or compiled on the fly its functionality is the same.

Suppose there are to files (Java class) which are used by a Java program will be loaded into the memory one by one on a demand basis. JVM will not check for the accessibility of the files before executing the code. In case if it is not able to find the required file in between the execution of the program, JVM will throw an exception saying that it has not found the file and exit the program from that point. The program will not complete successfully.

Although the other programming languages which existed before the origin of Java were as good and user friendly to the professional programmers, they still expected something advance to come up with all those features which were definitely the cause of worry to them with respect to the security of their code and

this thought gave birth to a revolution which discovered another Programming Language-JAVA with the features to ensure the security and the portability of their programs.

## FEATURES OF JAVA

Developing your applications using the Java programming language results in software that is portable across multiple machine architectures, operating systems, and graphical user interfaces, secure, and high performance.

Not only the security, but there were few more areas which were taken care by this Programming Language normally identified as the JAVA-BUZZWORDS. These words define the additional features and considerations which gave JAVA-complete success and acceptance from the programmers from the software world.

Together, the above requirements comprise quite a collection of buzzwords. So, let's examine some of them and their benefits before going on.

- Simple and Object oriented
- Secure and Portable
- Robust
- Multithreaded and Architecture-neutral
- Interpreted and High performance
- Distributed and Dynamic

Although these words are self-explanatory in them. We will take a look on each one of them in our tutorial below.

### Simple and Object-Oriented

As mentioned above while discussing the overview of Java we understood that Java got its origin as a result of the deep study of the pre-existing Programming languages like C and C++. It makes very easy for any professional programmer to have a clear understanding of Java and its functionality if he has basic knowledge

of C++ and the OOPS (Object Oriented programming) concepts. And hence the fundamental concepts of Java technology can be grasped quickly and the programmers can be productive from the very beginning as its look and feel makes it comfortable to the programmers as the beginners of Java Too. Not only this, the Founders of JAVA made sure that although Java is originated from the basics of the pre-existing Programming Languages it still avoids the features of those languages which were confusing and were not accepted happily by the Professionals. The Java programming language is designed to be object oriented from the ground up. After thirty years of regular exercise finally the Object technology got the acceptance in the programming mainstream. The Object-Oriented Concepts made it possible to function within increasingly complex, network-based environments, and so it can be concluded that Java technology provides a clean and efficient object-based development platform to the programmers.

## Secure & Portable

As discussed previously, under the head JVM (Java Virtual machine), the output which we get from the Java Complier is not in the form of directly 'Executable Code' but it is in the form of BYTE CODES., Byte code is nothing but the set of instructions which are executed by the Java Run time Environment, the JVM. Irrespective of all the other programming languages which are complied, java is actually interpreted by the JVM and hence makes it very much safe and secure to be downloaded through the internet as JVM makes sure that the Java Program is in its complete control and do not affect anything outside its environment. Since the code gets converted into the Byte Code by the JVM, it gives huge amount of portability to Java as it can run on any platform in any environment provided, they have the JVM available on it. The Byte codes will always be the same in spite of having variations in the JVM with respect to the environment and this is one of the other very strong supportive features of JAVA.

## Robust

Java is considered to be a very robust Programming language ass it ensures that the Java Programs run easily and successfully on the variety of environments and platforms. Since Java is a strictly typed language it helps the programmer to reduce the chances of making the run time errors and also handling them within the programs itself which normally cannot happen in other programming languages.

## Multi-threaded and Architecture Neutral

Multithreading is another feature of Java which lets programmer opt for Java as the programming language as it facilitates the programmer to do various tasks within one program itself without having issues of conflicts between the tasks. As it has been discussed earlier in the sections above, that Java Program gets converted into Byte Code which is very much platform independent it makes Java as highly Architecture-neutral. And hence it can be stated that Java has successfully achieved "write once; run anywhere, anytime, forever." which was the basic goal of the inventors of Java Programming Language.

## Interpreted and High Performance

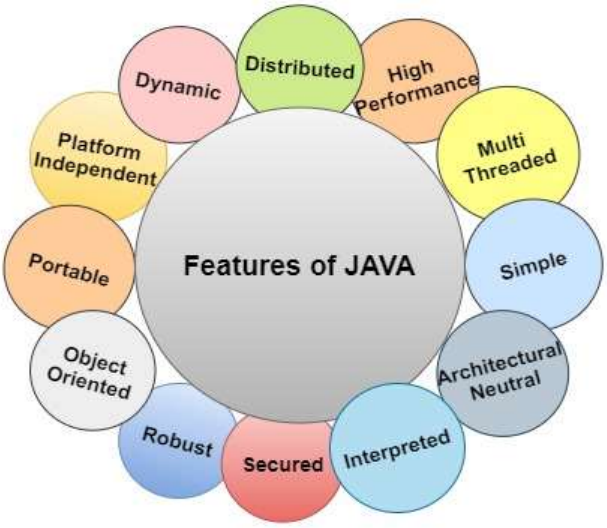
Java programs get converted into Byte code which is easily interpreted by the JVM on any machine and in any environment and although it gets interpreted by the JVM it is not the case that the Java Code can't be complied into the native machine code and it is taken care by the JIT (Just –In-Time) Complier which is available with the JVM and thus ensures very high Performance of the language.

## Distributed and Dynamic

Since, Java was designed with the aim of making it accessible over the internet, providing it the feature of being called as Distributed programming language was very important as it had to handle the TCP/IP protocols of the internet. Java technology and its runtime environment have combined them to produce a flexible and powerful programming system.

Since the byte codes are loaded in small application forms as applets it becomes mandatory for Java programs to carry the sufficient amount of information related to the objects which are executed at the run time so as to provide verification with respect to that particular object hence giving it the feature of being called as Dynamic Language too.

These above-mentioned features can be easily termed as the major wings of the Java Programming language which has given it the open acceptance of the professions of the programming World.



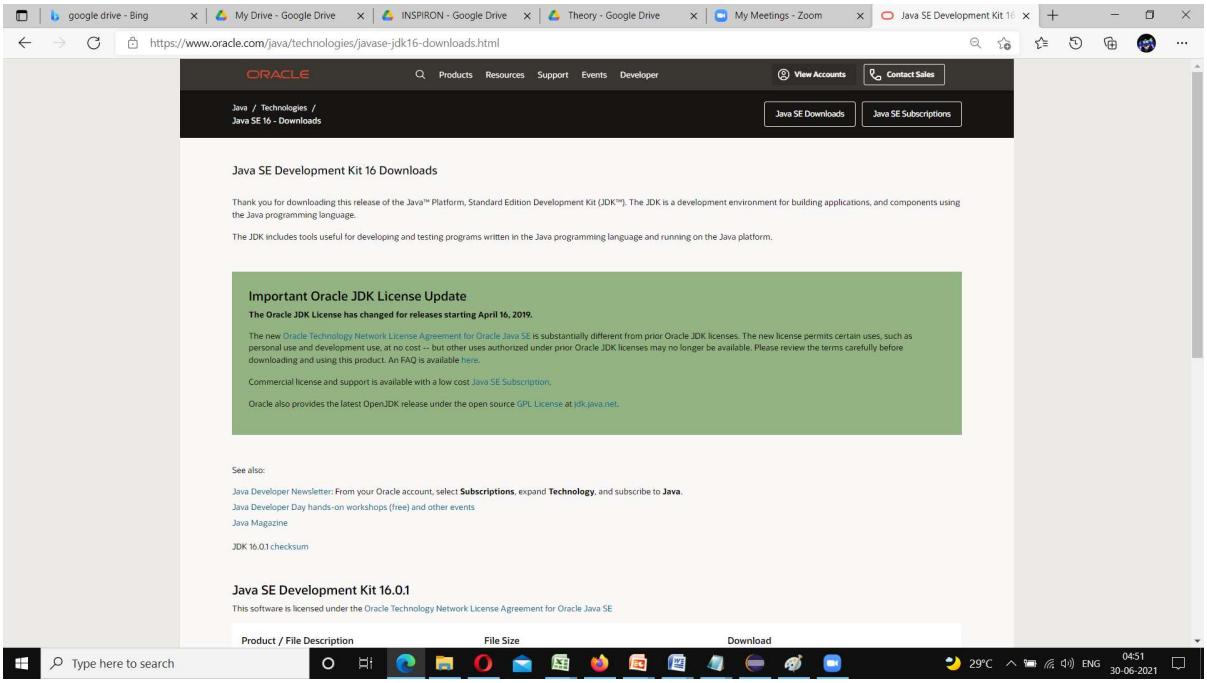

## INSTALLATION, HOW TO SETUP PATH?

In order to execute java programs and application one needs to install Java Runtime Environment that has all supporting tools like java compiler, java

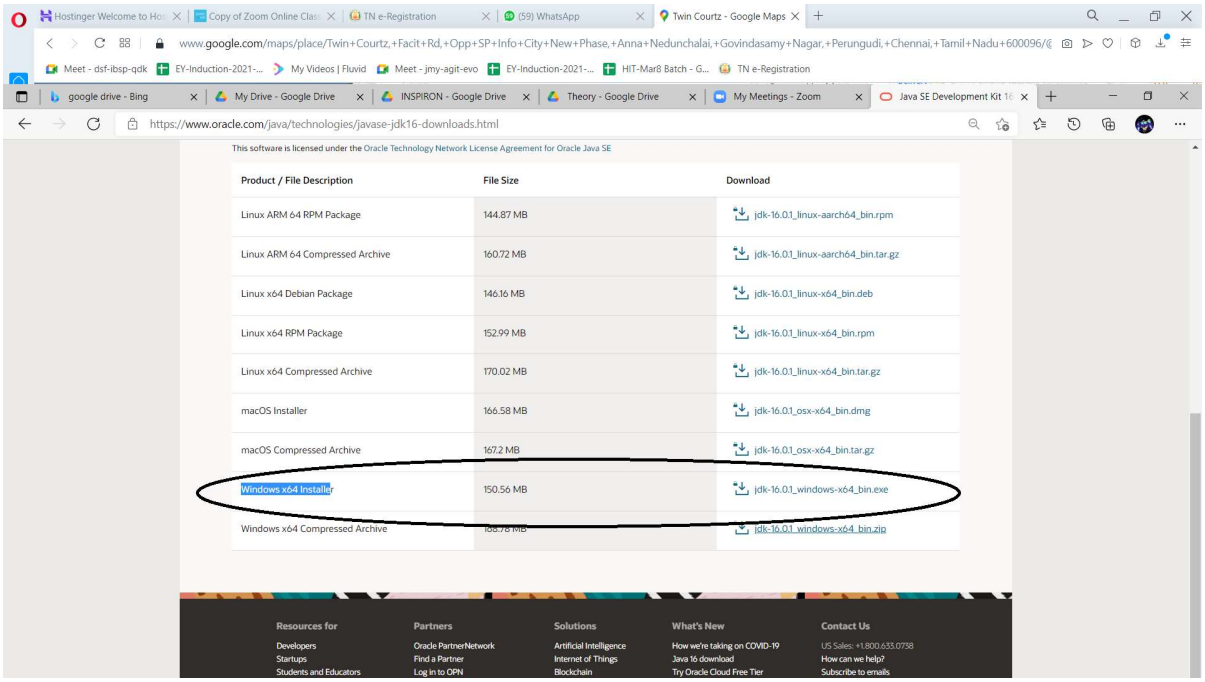
debugger, tools to execute java programs. In order to install JDK (Java Development Toolkit) on the computer you need to download the JDK from Java Sun Microsystems.

Please follow the step to download JDK 1.5 from Sun Microsystems for Windows platform.

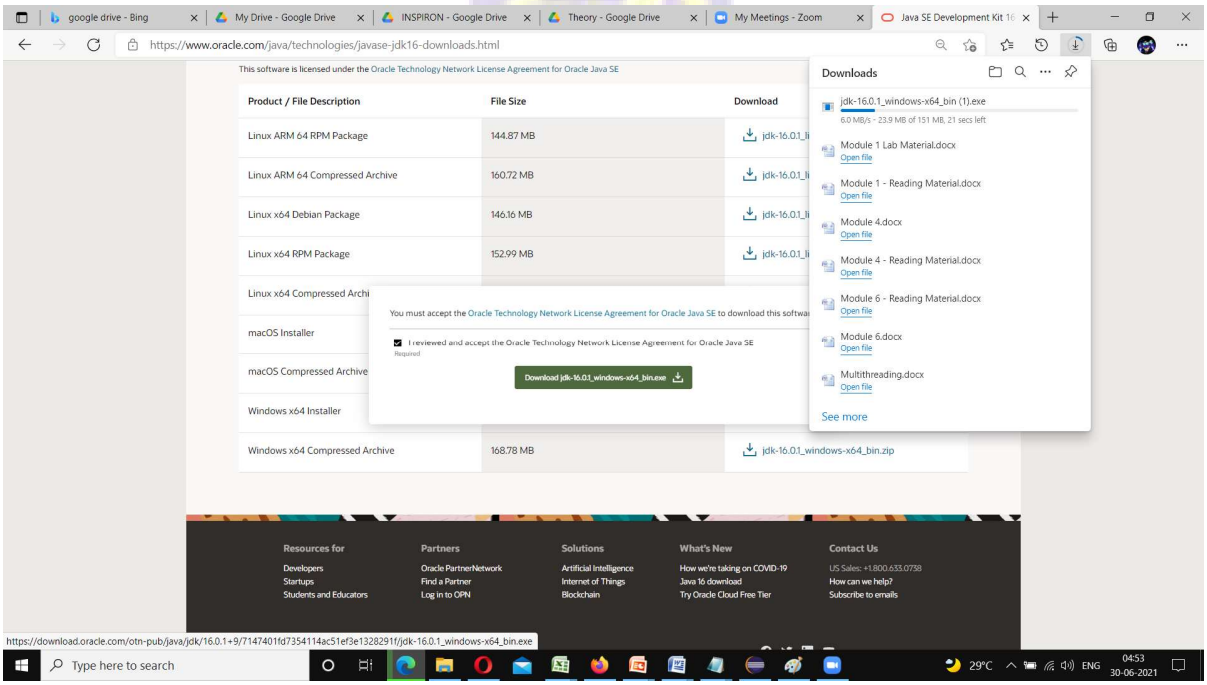
1. Enter the below URL in your browser.
2. https://www.oracle.com/java/technologies/javase-jdk16-downloads.html
3. You will find a screen like below.




2. Click on the link which is highlighted in the circle




3. It will immediately start downloading.

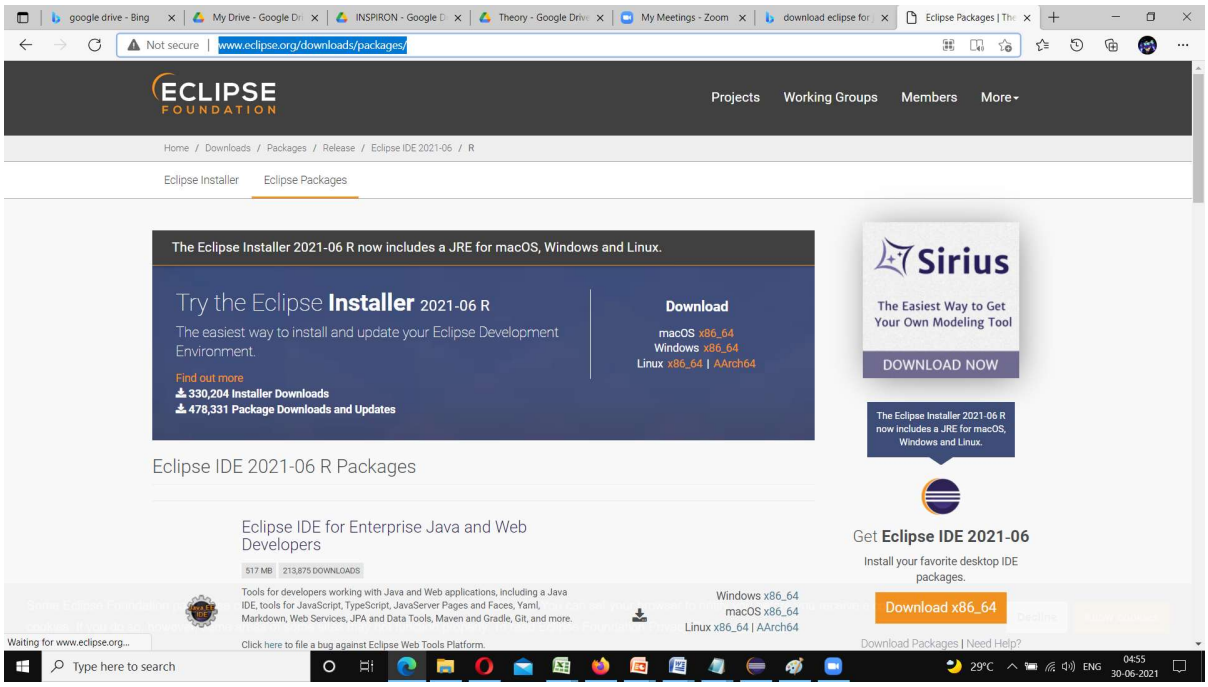


7. Once it is downloaded on your system you have to install it on your PC/laptop. Just follow the instructions given during the installation. Believe me it is not difficult to install JDK.

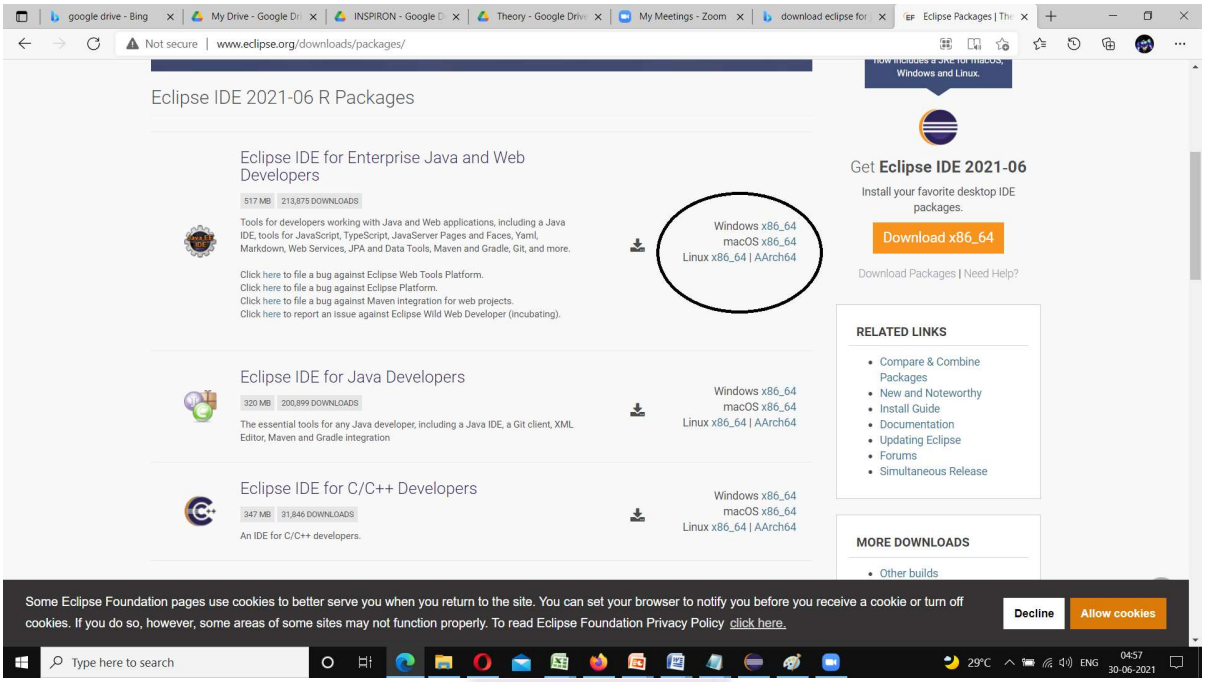
8. After installing JDK you will be able to work with JAVA

Now you download Eclipse

1. Enter the URL in your browser - http://www.eclipse.org/downloads/packages/



2.Click on the link specified in the circle, it will start downloading eclipse




Congratulations!!!, You have successfully installed java on your system.

## APPLICATIONS OF JAVA

- Mobile Applications
- Desktop GUI Applications
- Web-based Applications
- Enterprise Applications
- Scientific Applications
- Gaming Applications
- Big Data technologies
- Business Applications
- Distributed Applications
- Cloud-based Applications

# List of Java Keywords

| | | | | |
|---|---|---|---|---|
| boolean | byte | char | double | float |
| short | void | int | long | while |
| for | do | switch | break | continue |
| case | default | if | else | try |
| catch | finally | class | abstract | extends |
| final | import | new | instance of | private |
| interface | native | public | package | implements |
| protected | return | static | super | synchronized |
| this | throw | throws | transient | volatile |

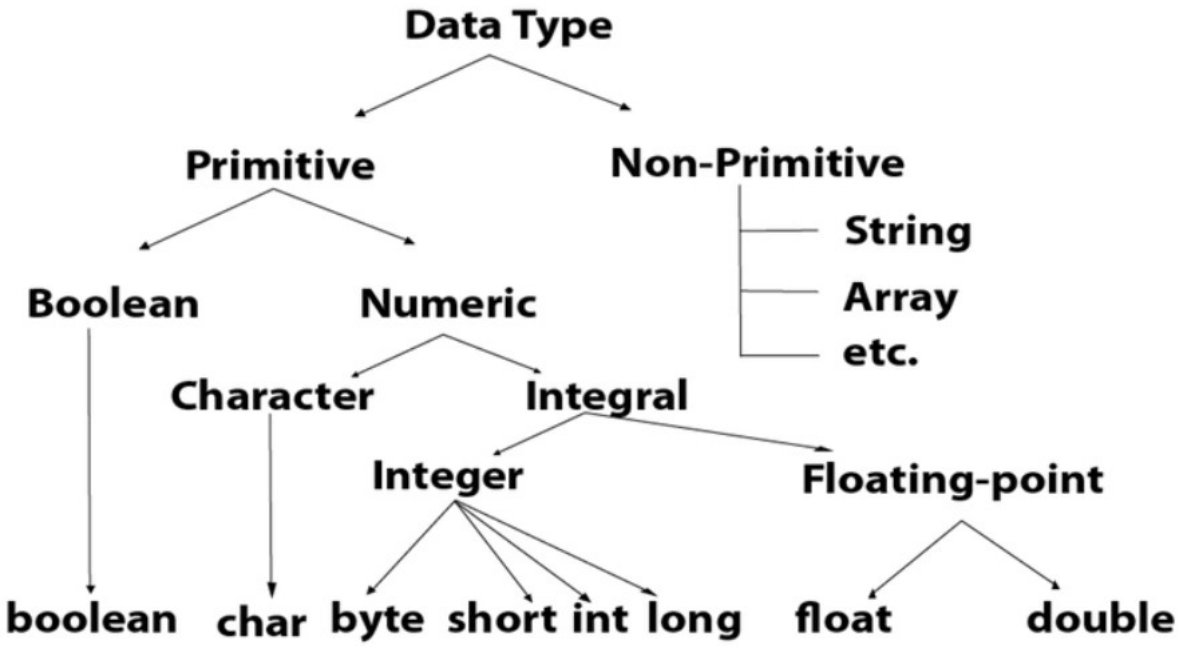
## DATA TYPES AND TYPE CASTING

Java is termed as strongly typed Programming which means that every variable which is defined in any line code of java programming has to be properly well defined and assigned a data type and this again makes it possible for java to maintain such complex robustness and security to the code and the data.

If we broadly categorize and classify these data types, we can define them as Eight primitive types of data types available to us in Java.

1. Four of them are integer types
2. Two are floating-point number types
3. One is the character type char, used for code units in the Unicode encoding scheme
4. One is a Boolean type for truth values

In most situations, the int type is the most practical. If you want to represent the number of inhabitants of our planet, you'll need to resort to a long. The byte and short types are mainly intended for specialized applications, such as low-level file handling, or for large arrays when storage space is at a premium.




Let's take these four categories in brief for our better understanding

1. Integers can be defined as byte, short, int, and long, which are for whole-Valued signed numbers.

   The int type is the most practical. If you want to represent the number of humans on earth you'll need to resort to a long. The byte and short types are mainly intended for specialized applications.

| Java Integer Types | | |
|---|---|---|
| Type | Storage Requirement | Range (Inclusive) |
| Int | 4 bytes | –2,147,483,648 to 2,147,483, 647 (just over 2 billion) |
| Short | 2 bytes | –32,768 to 32,767 |
| Long | 8 bytes | –9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 |
| Byte | 1 byte | –128 to 127 |

2. Floating-point numbers basically is used when we have the situation of getting our result or output in the form of decimals and not whole numbers as mentioned in case of Integers Data Types. This group thus includes float and double.

| Java Floating Types | | |
|---|---|---|
| Type | Storage Requirement | Range (Inclusive) |
| double | 64 bits | 4.9e324 to 1.8e+308 |
| float | 32 bits | 1.4e045 to 3.4e+038 |

3. Characters help defining the char, which represents symbols in a character Set, like letters and numbers.

| Character Escape Sequences | |
|---|---|
| Escape Sequence | Description |
| \uxxxx | Hexadecimal UNICODE character (xxxx) |
| \' | Single quote |
| \" | Double quote |
| \\ | Backslash |
| \r | Carriage return |
| \n | New line |
| \f | Form feed |
| \t | Tab |
| \b | Backspace |

4. Boolean data type is used to mention the variables which will only have the possibility of containing values either as True or False.

## VARIABLES AND ARRAYS

## Variable

The variable is the basic unit of storage in a Java program. You declare a variable by placing the type first, followed by the name of the variable. Here are some examples:

```
double salary;
double _salary;
int holidays;
int $holidays;
long distanceFromMoonToEart;
boolean done;
```

Notice the semicolon at the end of each declaration. The semicolon is necessary because a declaration is a complete Java statement. In addition, all variables have a scope, which defines their visibility, and a lifetime. General form of declaring variables:

**type identifier [ = value][, identifier [= value] ...] ;**

Where type identifier could be byte, int, long, boolean, char, short, float, double. Variable name cannot start with numeric character or special characters except "$" and "_". For valid example please refer above variable names. Let us also see some invalid variable names:

```
double 1salary;      // invalid variable name
double %salary;      // invalid variable name
int #holidays;       // invalid variable name
```

After you declare a variable, you must explicitly initialize it by means of an assignment statement—you can never use the values of uninitialized variables. For example, the Java compiler flags the following sequence of statements as an error.

```
byte z = 22;
```

Although the preceding examples have used only constants as initializers, Java allows variables to be initialized dynamically, using any expression valid at the time the variable is declared.

```
float a = 10.0f
float b = 3.4f
float i = a / b;
```

## Scope and Lifetime of Variables

Lifetime of a variable, that is, the time a variable is accessible during the execution, is determined by the context in which it is declared. We distinguish between lifetime of variables in three contexts:

Instance variables - members of a class and created for each object of the class. Every object of the class will have its own copies of these variables, which are local to the object. The values of these variables at any given time constitue the state of the object. Instance variables exist as long as the object they belong to exists.

Static variables - also members of a class, but not created for any object of the class and, therefore, belong only to the class. They are not associated with any object. They are created when the class is loaded at runtime, and exist as long as the class exists. Every object derived from the class will share static variables. In a multithreaded application static variable are discouraged because of their feature.

Local variable - declared in methods and in blocks and created for each execution of the method or block. After the execution of the method or block completes local variables are no longer accessible. Local variable should be initialized within the methods and blocks where they are created. Not doing so will cause java compilation error.

Create a Java Program

To create a simple java program, you need to create a class that contains the main method. Let's understand the requirement first.

---

The requirement for Java Hello World Example

For executing any java program, you need to

- Install the JDK if you don't have installed it,
- Set path of the jdk/bin directory. http://www.javatpoint.com/how-to-set-path-in-java
- Create the java program
- Compile and run the java program

---

## CREATING HELLO WORLD EXAMPLE

Let's create the hello java program:

```
class Simple{
    public static void main(String args[]){
     System.out.println("Hello Java");
    }
}
```

## PARAMETERS USED IN FIRST JAVA PROGRAM

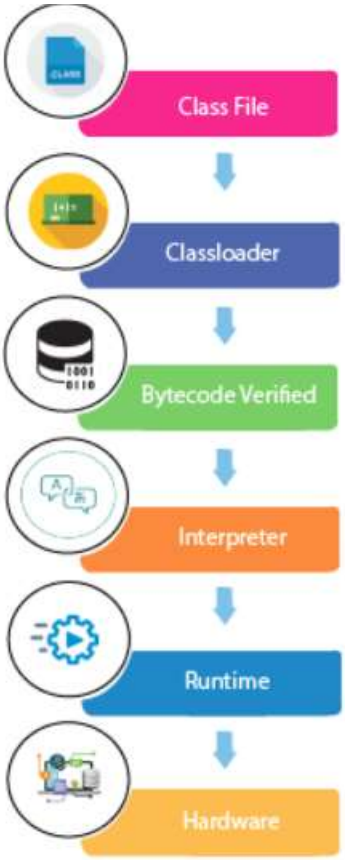
Let's see what is the meaning of class, public, static, void, main, String[], System.out.println().

- class keyword is used to declare a class in java.

- **public** keyword is an access modifier which represents visibility. It means it is visible to all.
- **static** is a keyword. If we declare any method as static, it is known as the static method. The core advantage of the static method is that there is no need to create an object to invoke the static method. The main method is executed by the JVM, so it doesn't require to create an object to invoke the main method. So it saves memory.
- **void** is the return type of the method. It means it doesn't return any value.
- **main** represents the starting point of the program.
- **String[] args** is used for command line argument. We will learn it later.
- **System.out.println()** is used to print statement. Here, System is a class, out is the object of PrintStream class, println() is the method of PrintStream class. We will learn about the internal working of System.out.println statement later.

## Details of Hello Java Program

we have learnt about the first program, how to compile and run the first java program. Here, we are going to learn, what happens while compiling and running the java program. Moreover, we will see some question based on the first program.



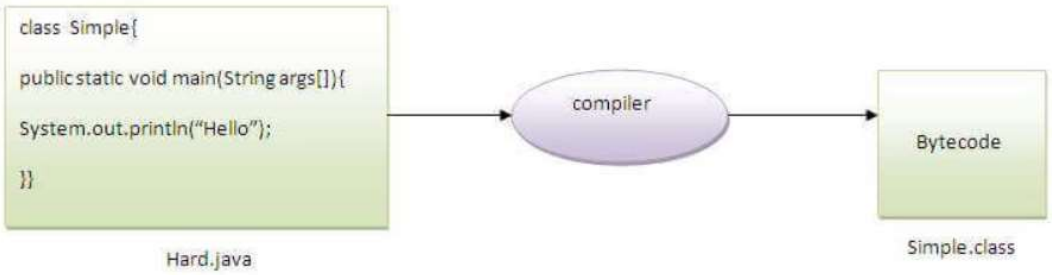

Classloader: is the subsystem of JVM that is used to load class files.

Bytecode Verifier: checks the code fragments for illegal code that can violate access right to obje

Interpreter: read bytecode stream then execute the instructio

## Q) Can you save a java source file by other name than the class name?

Yes, if the class is not public. It is explained in the figure given below:




**To compile:**

javac Hard.java

**To execute:**

java Simple