OBJECT™
TECHNOLOGIES

Introduction to Java

What will be covered

OBJECT TECHNOLOGIES

- History of Java
- Java Features
- First Program
- Java Installation
- Java versions
- Difference between JDK, JRE and JVM

C Programming :

- Most popular language
- Many other laguage prior to C like COBOL, PASCAL, BASIC, FORTRAN were lacking features in some area
- C dominated all other languages beacuse of following features
- English like language, easy to learn and understand
- Mid level language for developing system as well as application software
- structured way of programming.
- language designed for programmers

C++ Programming :

- The increasing complexity of programs has driven the need for better ways to manage that complexity.
- many projects were pushing the structured approach past its limits. To solve this problem, a new way to program was invented, called object-oriented programming (OOP)
- C++ extends C by adding object-oriented features.
- Because C++ is built on the foundation of C, it includes all of C's features, attributes, and benefits. This is a crucial reason for the success of C++ as a language
- We can simply call c++ as C with Classes

History of Java

OBJECT TECHNOLOGIES

Driving forces behind creation of java

Platform independent language :

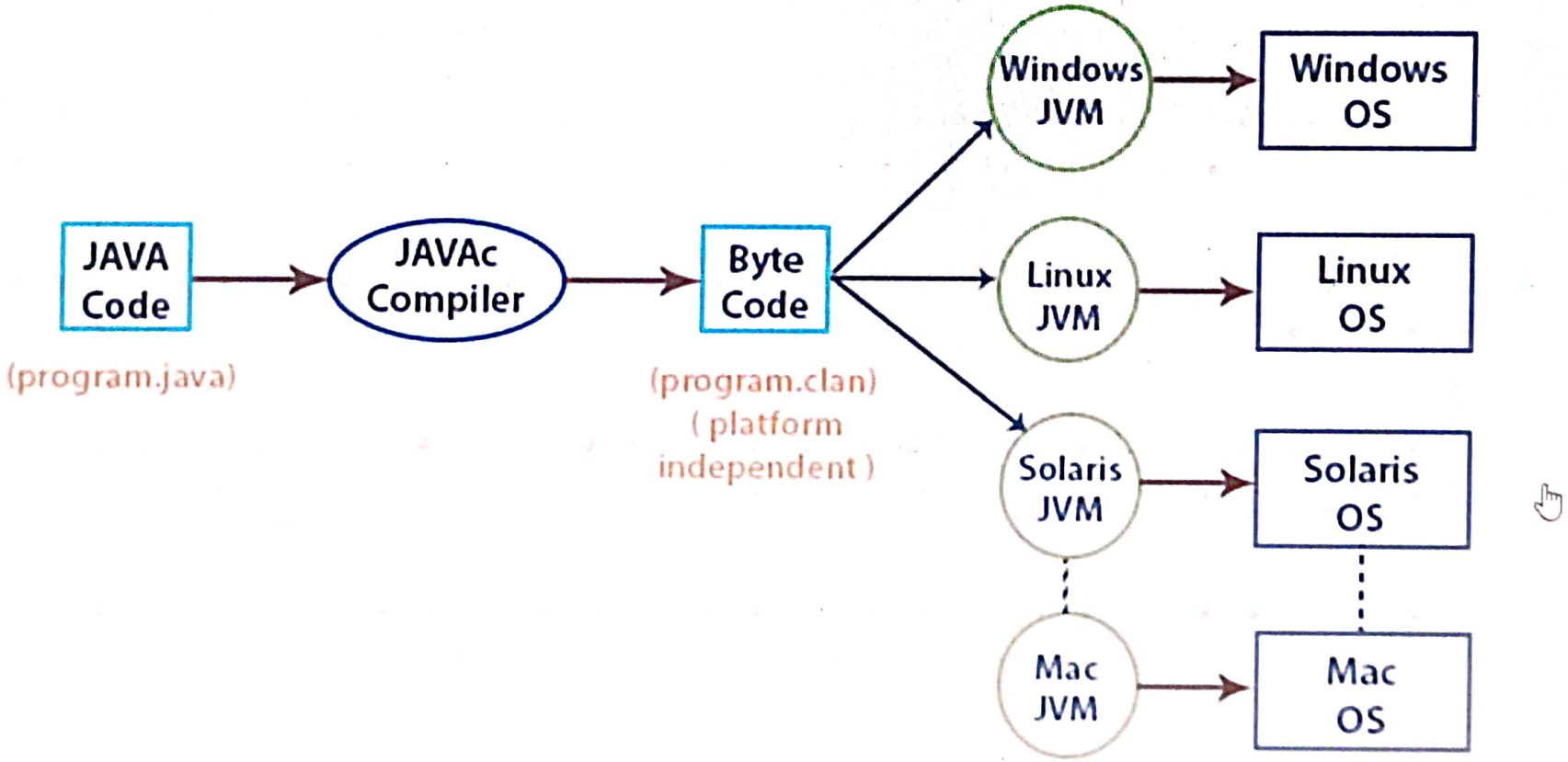
- the primary motivation was the need for a platform-independent (that is, architecture-neutral) language that could be used to create software to be embedded in various consumer electronic devices, such as microwave ovens and remote controls.
- Many different types of CPUs are used as controllers. C and C++ languages are designed to be compiled for a specific target(platform which is combination of OS and processor).
- Although it is possible to compile a C++ program for just about any type of CPU, atleast full C++ compiler targeted for that CPU is needed.
- Compilers are expensive and time-consuming to create.
- So a project is undertaken for developing platform independent, portable language as a easy and cost-efficient solution.

Portability

- Portability is a major aspect where same program can run on different types of computers and operating systems.
- Compilation model of C/C++ focuses on generating executable on a specific platform.
- As compared with this, java compilation model is different.
- It focuses on generating byte code in the form of .class file which is platform independent.
- Bytecode is someting which is neither readable like source code nor executable like executable code
- This byte code is interpreted on all the platforms with the help of specific JVM.
- JVM is platform dependent and the compiled code is platform independent
- This way prtabilty is achieved like write once, run everywhere

TechVidvan

Platform Independence in Java



Driving forces behind creation of java

emergence of the World Wide Web

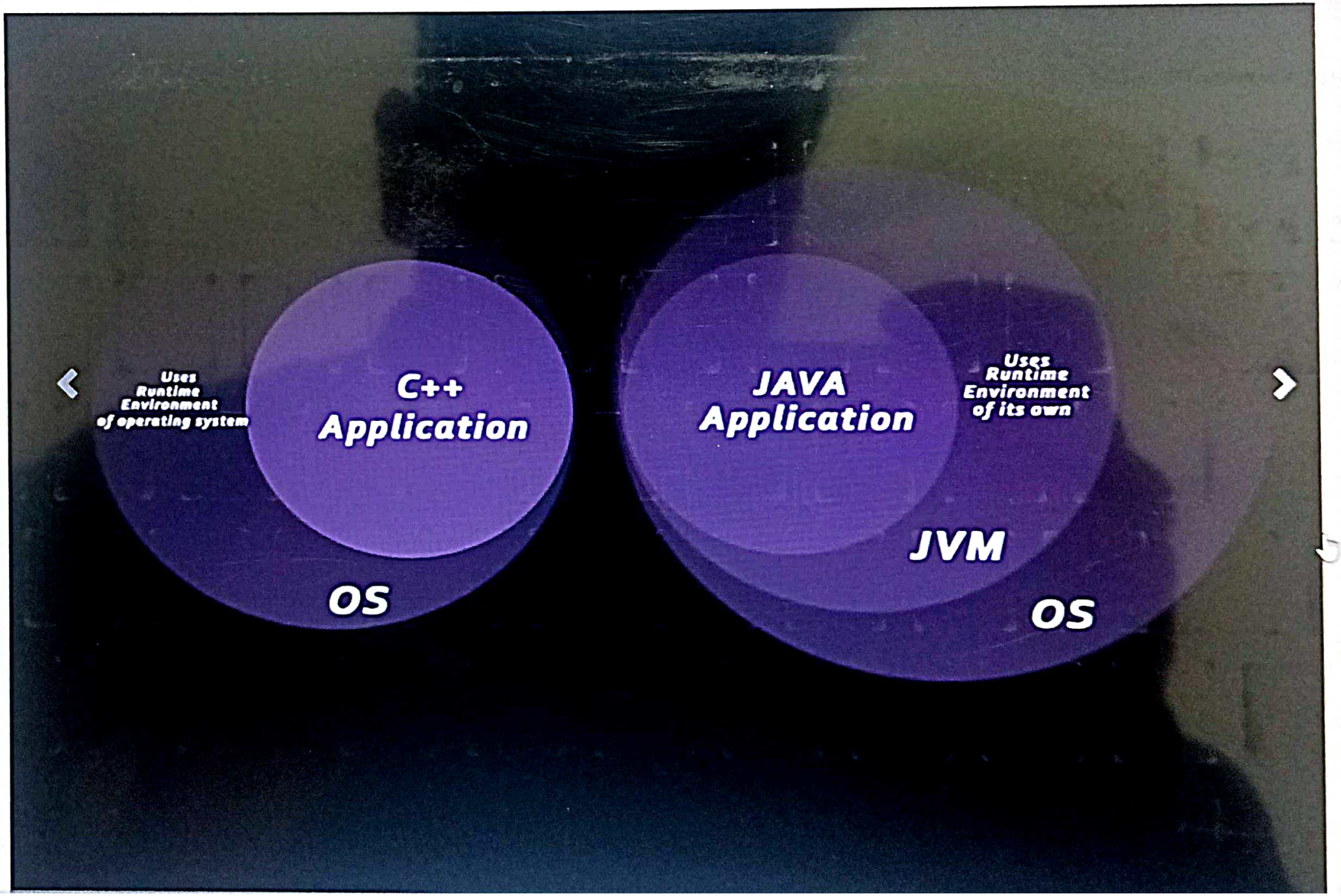
- It even demanded portable programs.
- the Internet consists of a diverse, distributed universe populated with various types of computers, operating systems, and CPUs.
- There was the need for platform-independent programs destined for distribution on the Internet
- It was a perfect response to the demands of the then newly emerging, highly distributed computing universe
- Java was to Internet programming what C was to system programming

- As you are likely aware, every time you download a "normal" program, you are taking a risk, because the code you are downloading might contain a virus, Trojan horse, or other harmful code.
- Such malicious code can cause its damage because it has gained unauthorized access to system resources. For example, a virus program might gather private information, such as credit card numbers, bank account balances, and passwords, by searching the contents of your computer's local file system.
- In order for Java to enable applets to be downloaded and executed on the client computer safely, it was necessary to prevent an applet from launching such an attack.
- Java achieved this protection by confining an applet to the Java execution environment and not allowing it access to other parts of the computer.
- Applets can not access client's file system but can access file system where it is hosted

Security

- As you are likely aware, every time you download a "normal" program, you are taking a risk, because the code you are downloading might contain a virus, Trojan horse, or other harmful code.
- Such malicious code can cause its damage because it has gained unauthorized access to system resources. For example, a virus program might gather private information, such as credit card numbers, bank account balances, and passwords, by searching the contents of your computer's local file system.
- In order for Java to enable applets to be downloaded and executed on the client computer safely, it was necessary to prevent an applet from launching such an attack.
- Java achieved this protection by confining an applet to the Java execution environment and not allowing it access to other parts of the computer.
- Applets can not access client's file system but can access file system where it is hosted

Security


Uses
Runtime
Environment
of operating system
C++
Application
OS
JAVA
Application
Uses
Runtime
Environment
of its own
JVM
OS

Features of Java

OBJECT TECHNOLOGIES

- Simple
- Object Oriented
- Platform independent
- Architecture Neutral
- Compiled and interpreted
- High Performance
- secure
- robust
- multithreaded
- distributed
- dynamic

Java Installation

OBJECT TECHNOLOGIES

- **Step 1 : Download JDK**
- Go to Oracle website. Select appropriate version of JDK to install and select the installable based on client machine's operating system. Acept the license for download to start
- **Step 2 : Install JDK**
- Just double click the installable file. It makes you go through the wizard. Click on next button for completing installation
- **Step 3 : Set the path**
- Java's installation by default takes the path C:\Program Files\Java. In this path you will find new folder created as per the selected version of JDK e.g jdk1.8.0_121. Check the bin folder in this installation folder for existence of different java tools.
- Open command prompt in Windows

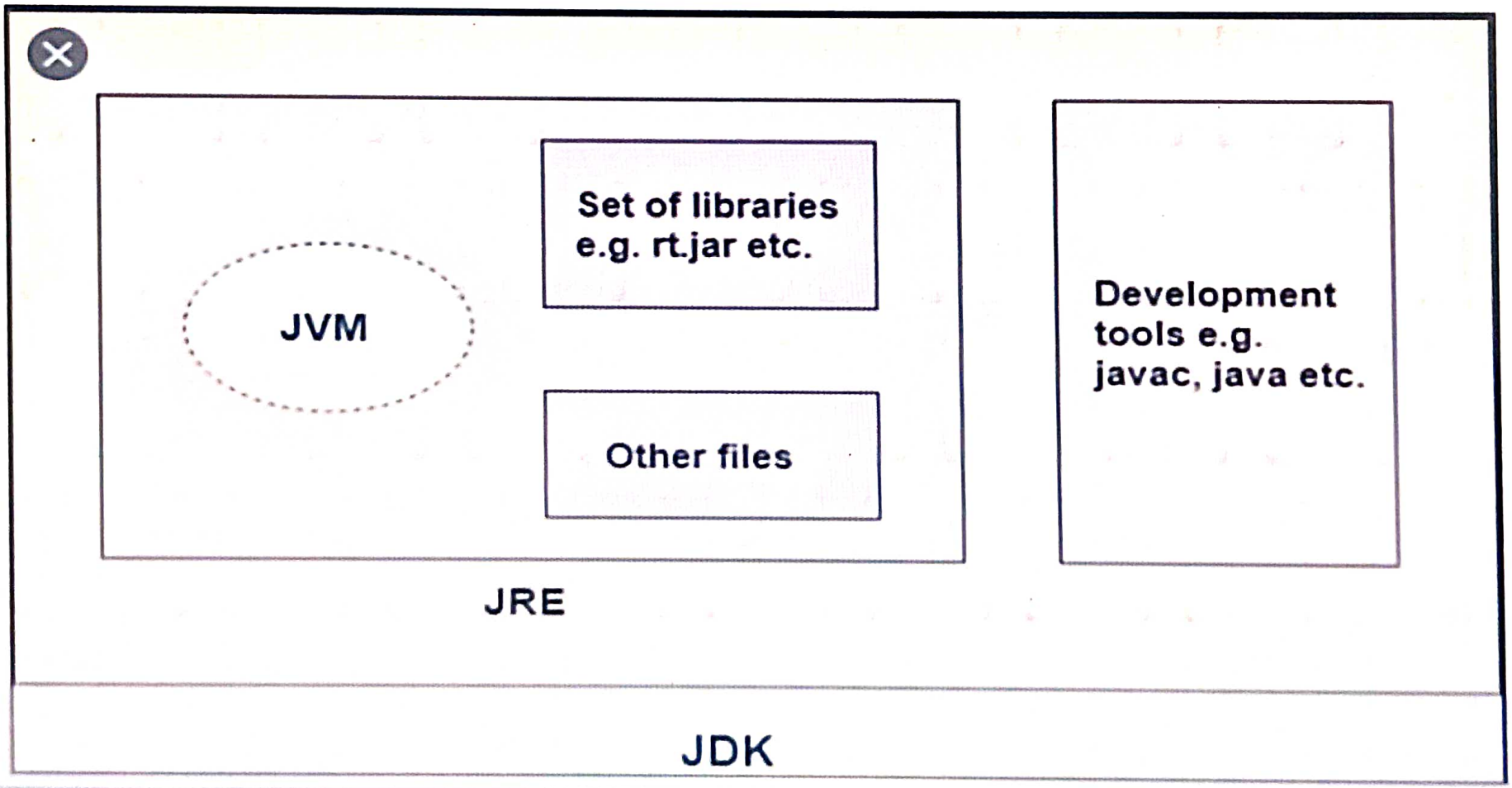
- Copy the path of jdk/bin directory where java located (C:\Program Files\Java\jdk_version\bin)
- Write in the command prompt: SET PATH=C:\Program Files\Java\jdk_version\bin
- Remember this path will be affected temporarily. For permanently setting the path modify Environment Variables available in advanced system settings
- **Step 4 : Verify installation**
- Open command prompt
- Execute the command javac -version
- You should get the display of respective version of java installed

Java versions

- As of March 2021, Java 16 is the latest released Java version.
- Legacy projects in companies are often stuck with using Java 8
- Java is special in this regard, as it is extremely backwards compatible. This means that your Java 5 or 8 program is guaranteed to run with a Java 8-16 virtual machine
- Java 8 serves a good base for all language features and then learn what additional features came in Java 9-14

JDK, JRE and JVM

- JDK - Required by deveopers for developing and testing the aplication
- JRE - It is needed by client machine to run existing java application(already compiled)
- JVM - It is subset of JRE, execution environment for bytecode



Java Features

1. **Simple**: Java is easy to learn and its syntax is quite simple, clean and easy to understand. The confusing and ambiguous concepts of C++ are either left out in Java or they have been re-implemented in a cleaner way. It is easy to migrate to java language for C/C++ programmers.
2. **Object Oriented**: In java, everything is an object which has some data and behavior. All methods should be part of java class. Variables can be declared only within the class or within the method. Primitive types, such as integers, are kept as high-performance nonobjects.
3. **Robust**: Robust is strong or tough. There are two main reasons for program failure: memory management mistakes and mishandled exceptional conditions (that is, run-time errors). Java virtually eliminates these problems by managing memory allocation and deallocation for the programmer. Deallocation is completely automatic, because Java provides garbage collection for unused objects. Java provides excellent and easy object-oriented exception handling. In a well-written Java program, all run-time errors can—and should—be managed by your program.
4. **Platform Independent**: Java provides a mechanism that allows the same application to be downloaded and executed by a wide variety of CPUs, operating systems, and browsers. Output of a Java compiler is not executable code. Rather, it is bytecode. Bytecode is a highly optimized set of instructions designed to be executed by Java Virtual Machine (JVM), which is part of the Java Runtime Environment (JRE).
5. **Architectural Neutral**: Operating system upgrades, processor upgrades, and changes in core system resources can all combine to make a program malfunction.Java's goal was "write once; run anywhere, any time, forever." which has been achieved to a great extent.
6. **Secure**: Every time when a program is downloaded, it involves a risk, because the code you are downloading might contain a virus, or other harmful code. At the core of the problem is the fact that malicious code can cause its damage because it has gained unauthorized access to system resources. Java achieved this protection by enabling you to confine an application to the Java execution environment and prevent it from accessing other parts of the computer.
7. **Multi Threading**: One of the ways to create a performance efficient application is to utilize multithreading. Java supports multithreading through library classes and interfaces. Java Thread allows us to create a lightweight process that executes some tasks. Java runtime will take care of creating machine-level instructions and work with OS to execute them in parallel.
8. **Compiled and Interpreted** : Java programs are compiled into an intermediate representation called Java bytecode. Bytecode is neither readable like source code nor executable like exe. Java virtual machine interprets this byte code during the execution of java programs.
9. **High Performance**: Java bytecode is designed in such a way that it would be easy to translate directly into native machine

9. **High Performance:** Java bytecode is designed in such a way that it would be easy to translate directly into native machine code by using a just-in-time compiler. The Just-In-Time (JIT) compiler is a component of the Java™ Runtime Environment that improves the performance of Java applications at run time.
10. Distributed : Java is designed for the distributed environment of the Internet because it handles TCP/IP protocols. In fact, accessing a resource using a URL is not much different from accessing a file. Java also supports Remote Method Invocation(RMI).
11. Dynamic : A Java program can read strings from external files, register those strings as classes, load those classes via custom class loaders, create proxies, serialize them, and so on, all at runtime.