Object Oriented Programming in Java :

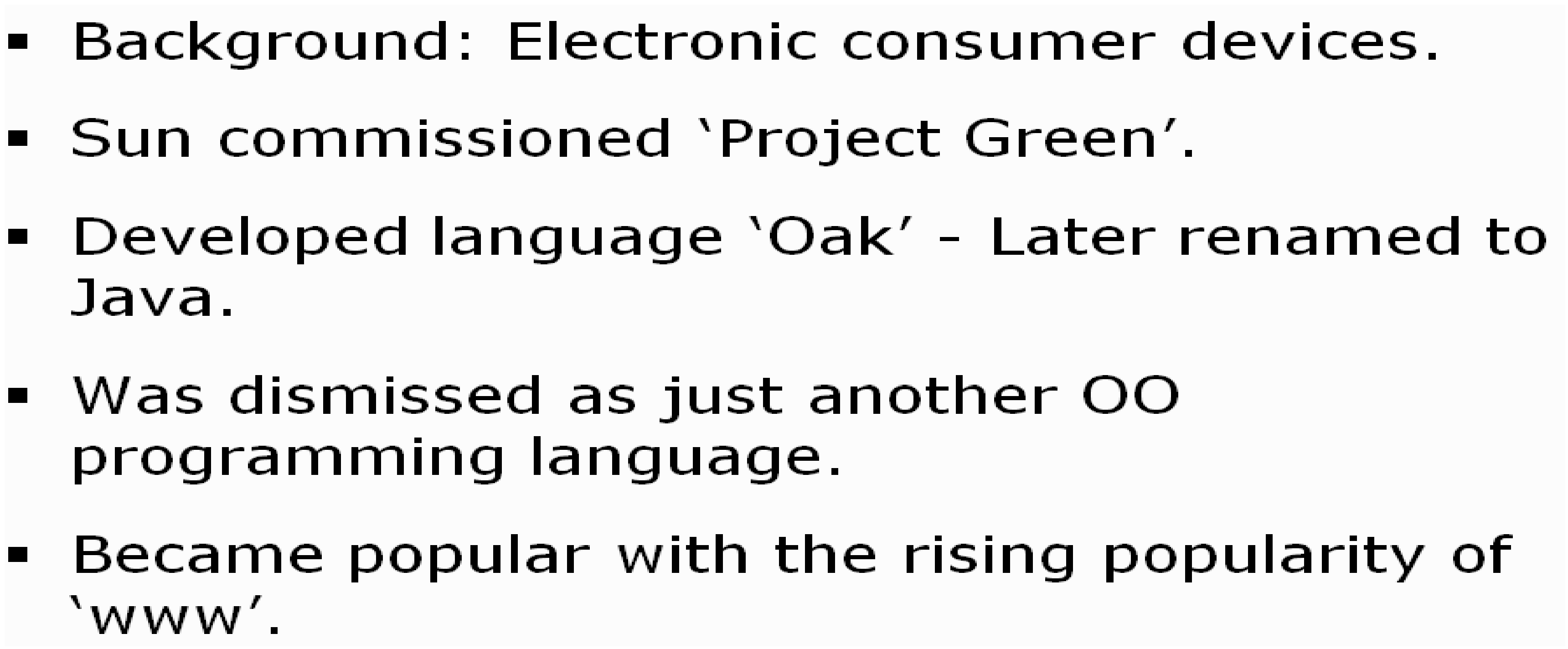
Java characteristics, Classes and Objects, Methods and Constructors. Information hiding: access modifiers, Static keyword: class variables and instance variables, Class methods and instance methods.

Java

- Java is a multi-platform, object-oriented, and network-centric language.
- It is among the most used programming language

History of Java

- James Gosling developed the Java platform at Sun Microsystems, and the Oracle Corporation later acquired it.



- Background: Electronic consumer devices.
- Sun commissioned ‘Project Green’.
- Developed language ‘Oak’ - Later renamed to Java.
- Was dismissed as just another OO programming language.
- Became popular with the rising popularity of ‘www’.

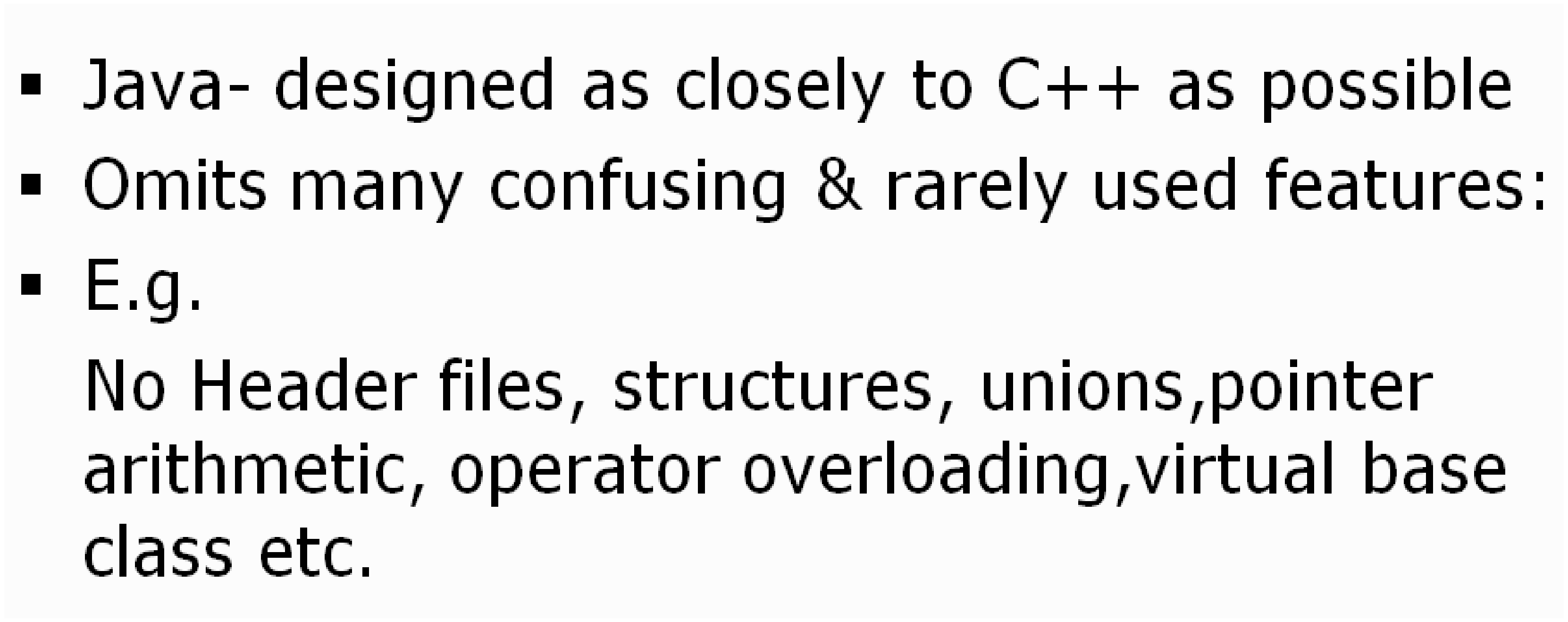
What is Java used for?

- It is used for developing Android Apps
- Helps you to create Enterprise Software
- Wide range of Mobile java Applications
- Scientific Computing Applications
- Use for Big Data Analytics
- Java Programming of Hardware devices
- Used for Server-Side Technologies like Apache, JBoss, GlassFish, etc.

Features of Java

- Simple
- Object-Oriented
- Platform independent
- Secured
- Robust
- Multithreaded
- Distributed
- Dynamic

- Simple
  - Java is very easy to learn,
  - syntax is simple, clean and easy to understand.



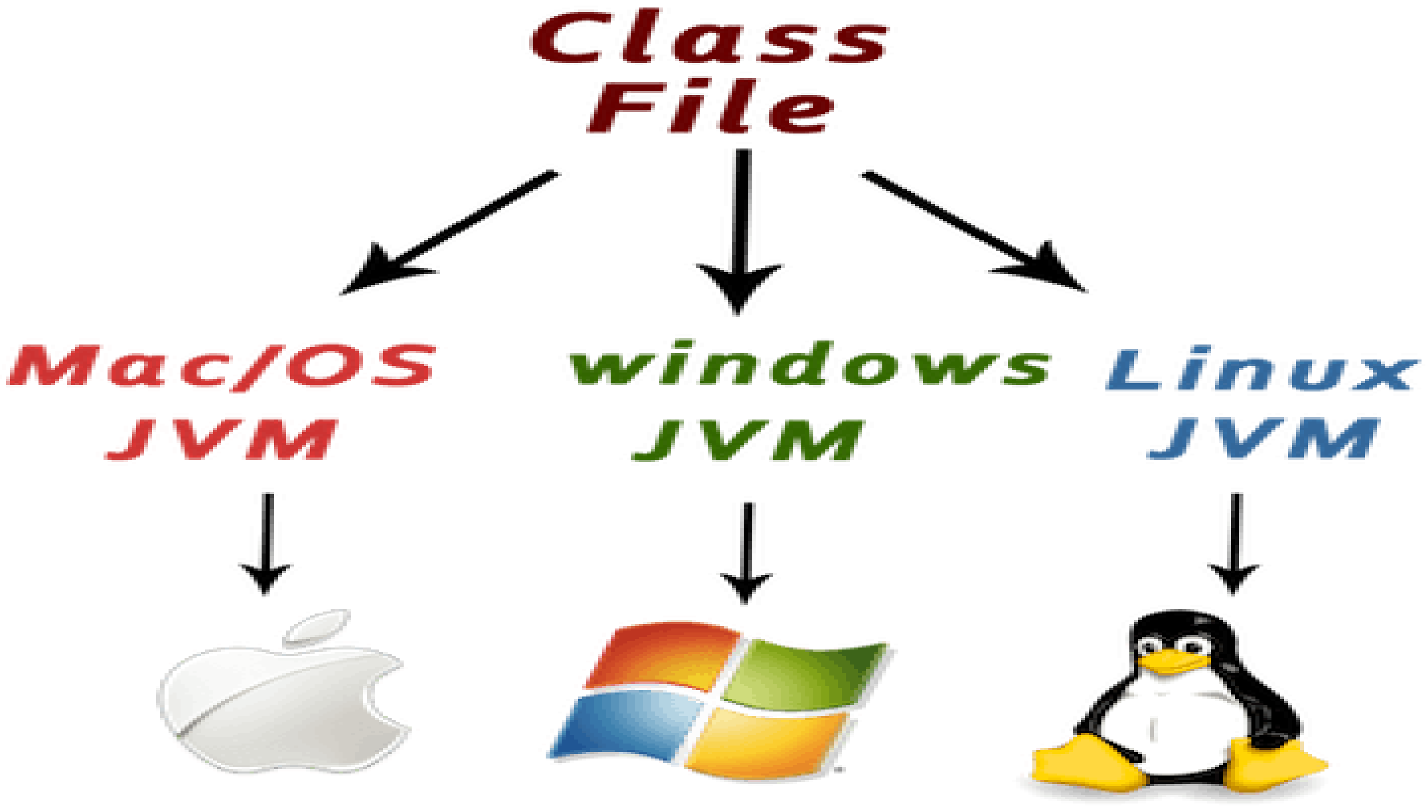
- Java- designed as closely to C++ as possible
- Omits many confusing & rarely used features:
- E.g.

  No Header files, structures, unions,pointer arithmetic, operator overloading,virtual base class etc.

Contd..

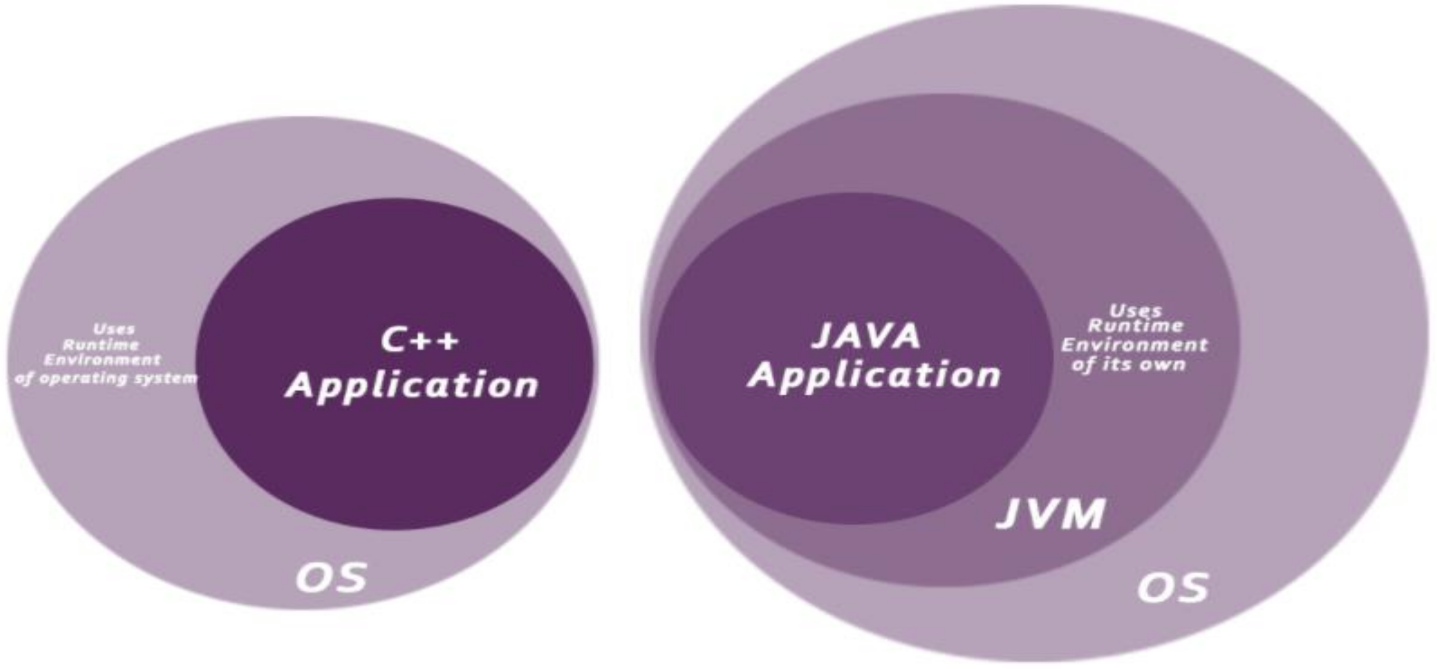
- Platform Independent
  - A platform is the hardware or software environment in which a program runs.
  - C, C++, etc. which are compiled into platform specific machines
  - Java is a write once, run anywhere language.(WORA)
  - Java code is compiled by the compiler and converted into bytecode.
  - This bytecode is a platform-independent code because it can be run on multiple platforms,

Contd…



contd..

- Secure
  - Java is best known for its security.
  - **Java Programs run inside a virtual machine**


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

Contd…

- Robust
  - Robust simply means strong.
- Java is robust because:
  - It uses strong memory management.
  - No pointers that avoids security problems.
  - automatic garbage collection
  - exception handling
  - Developers doesn't worry about –
    - Bad pointer
    - Memory allocation errors and memory leakage

Contd…

- Distributed
  - it facilitates users to create distributed applications in Java.
  - RMI and EJB are used for creating distributed applications.
  - This feature of Java makes us able to access files by calling the methods from any machine on the internet.

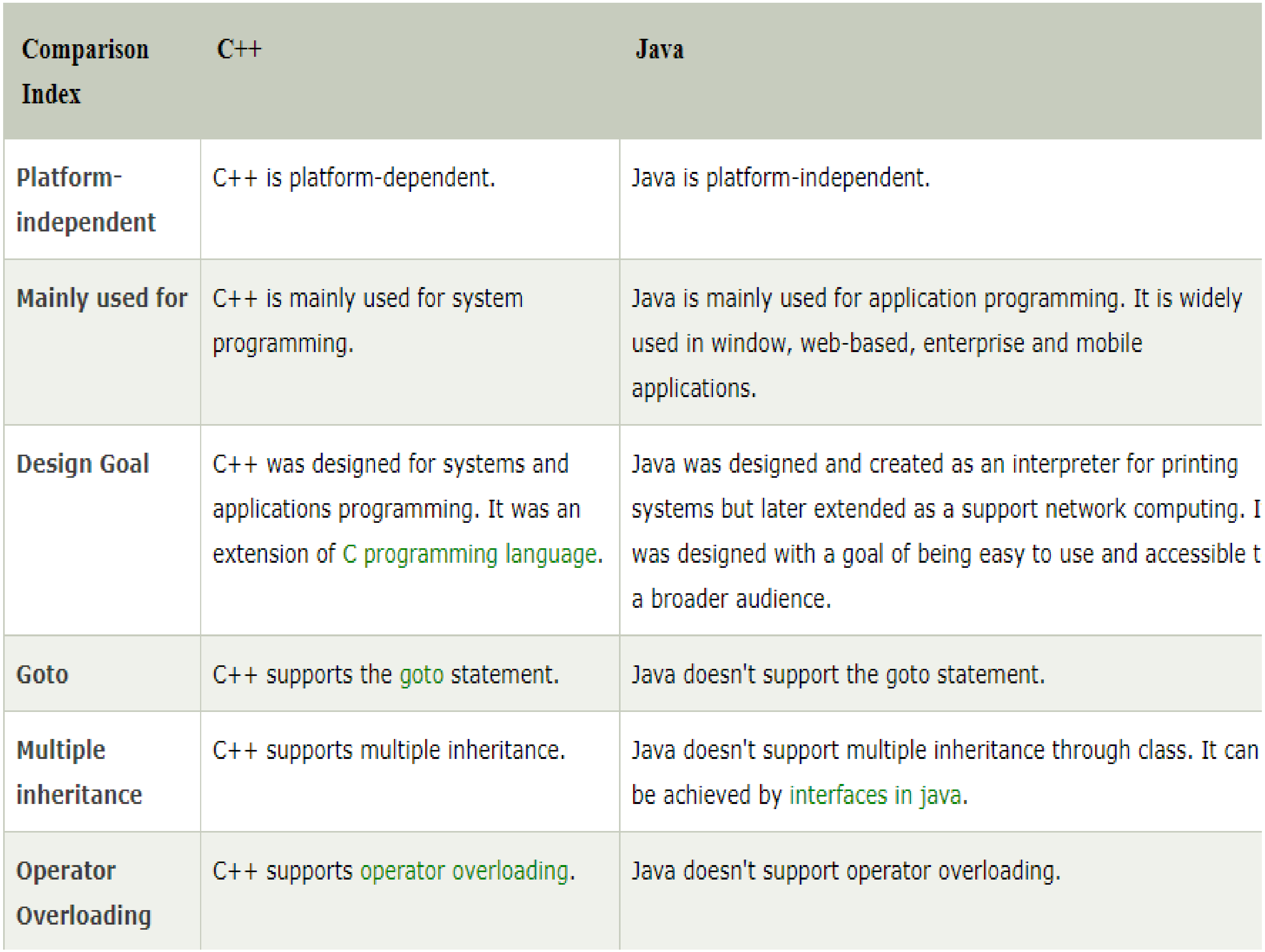
Contd…

- Multi-threaded
  - A thread is like a separate program, executing concurrently.
  - We can write Java programs that deal with many tasks at once by defining multiple threads.
  - The main advantage of multi-threading is that it doesn't occupy memory for each thread.
  - It shares a common memory area. Threads are important for multi-media, Web applications, etc.

Contd…

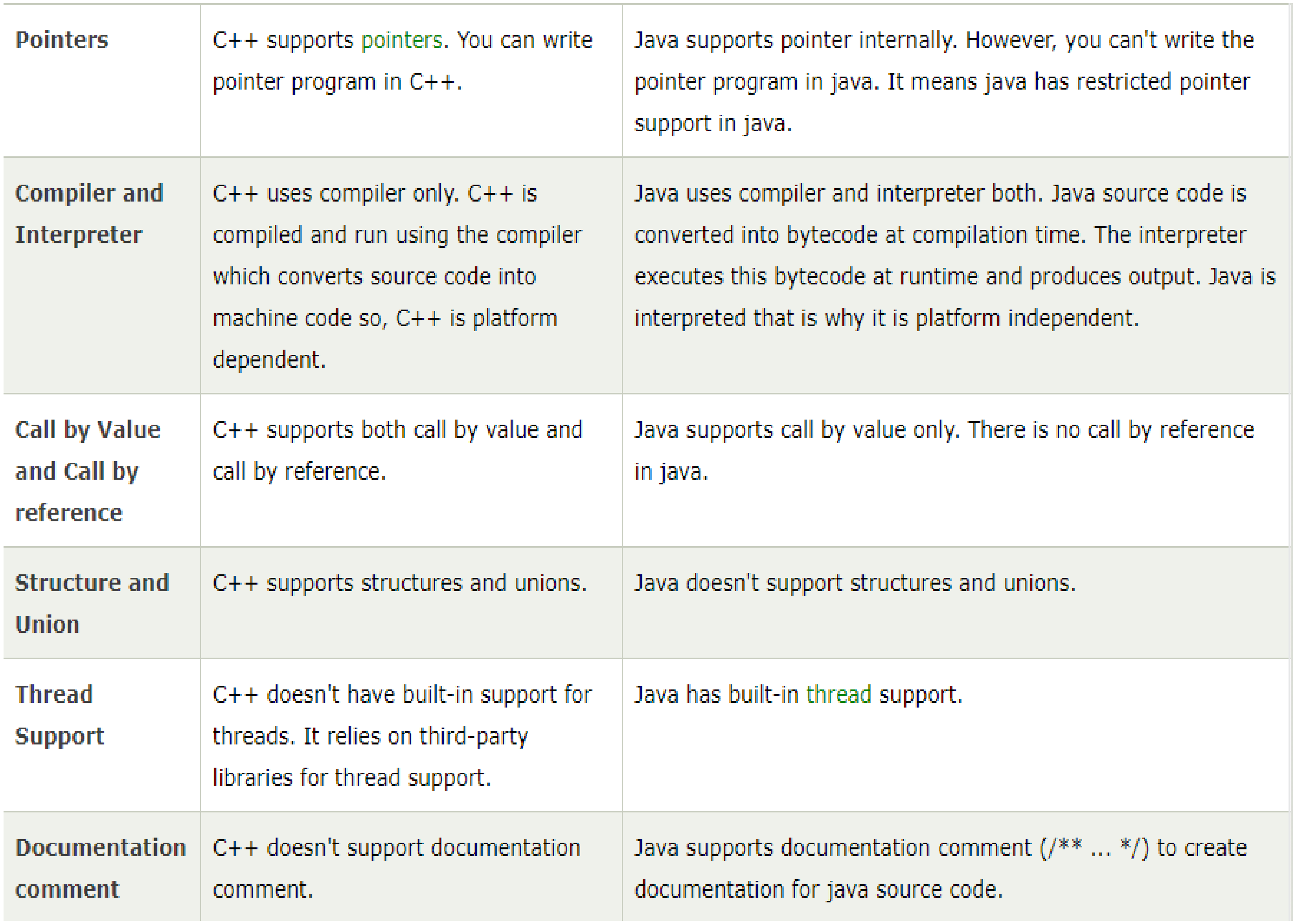
- Dynamic
  - Java is a dynamic language.
  - It supports dynamic loading of classes. It means classes are loaded on demand.
  - It also supports functions from its native languages, i.e., C and C++.
  - Java supports dynamic compilation and automatic memory management (garbage collection).

C++ vs Java



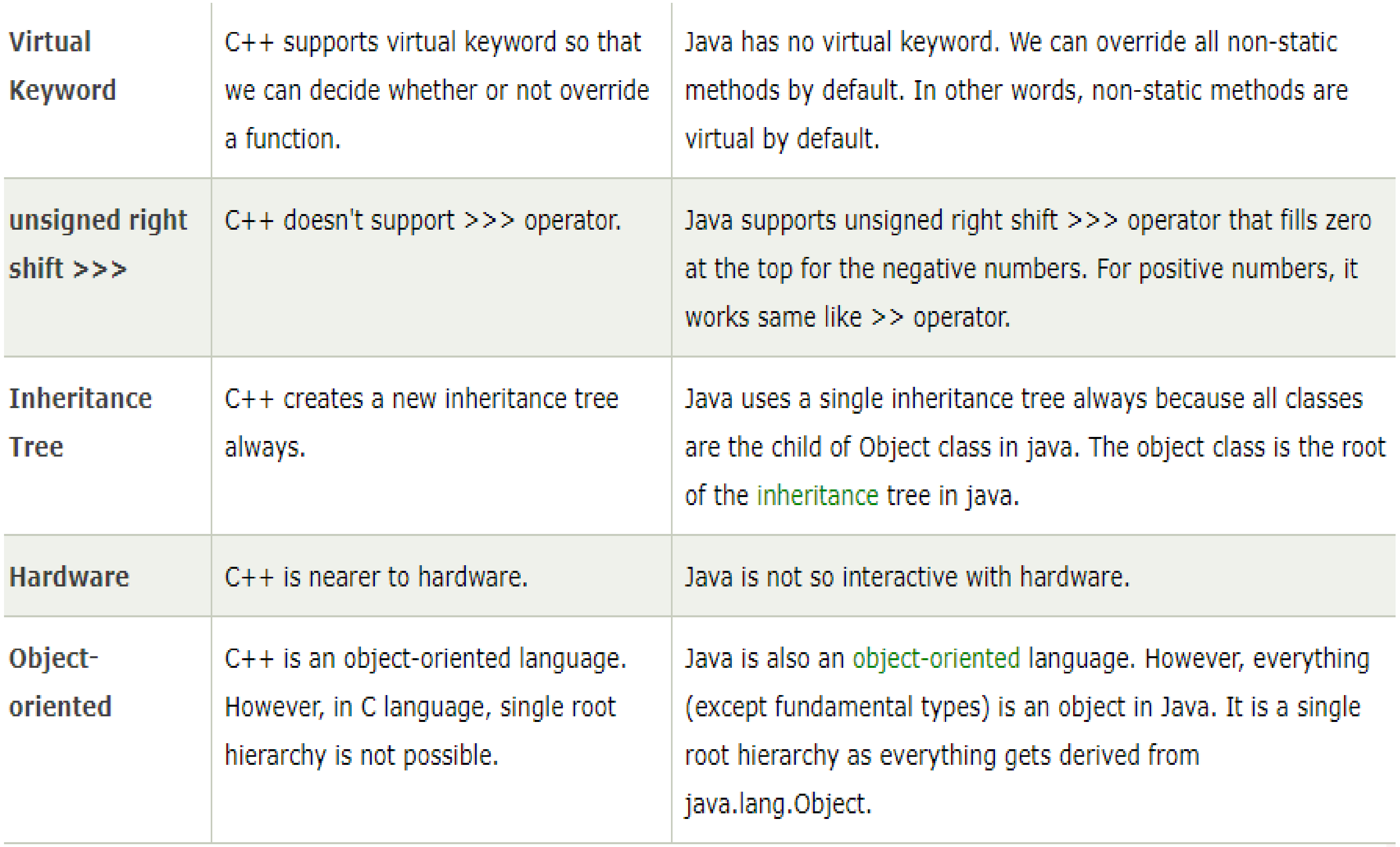
Comparison Index	C++	Java
Platform-independent	C++ is platform-dependent.	Java is platform-independent.
Mainly used for	C++ is mainly used for system programming.	Java is mainly used for application programming. It is widely used in window, web-based, enterprise and mobile applications.
Design Goal	C++ was designed for systems and applications programming. It was an extension of C programming language.	Java was designed and created as an interpreter for printing systems but later extended as a support network computing. I was designed with a goal of being easy to use and accessible t a broader audience.
Goto	C++ supports the goto statement.	Java doesn't support the goto statement.
Multiple inheritance	C++ supports multiple inheritance.	Java doesn't support multiple inheritance through class. It can be achieved by interfaces in java.
Operator Overloading	C++ supports operator overloading.	Java doesn't support operator overloading.

Contd…



Pointers	C++ supports pointers. You can write pointer program in C++.	Java supports pointer internally. However, you can't write the pointer program in java. It means java has restricted pointer support in java.
Compiler and Interpreter	C++ uses compiler only. C++ is compiled and run using the compiler which converts source code into machine code so, C++ is platform dependent.	Java uses compiler and interpreter both. Java source code is converted into bytecode at compilation time. The interpreter executes this bytecode at runtime and produces output. Java is interpreted that is why it is platform independent.
Call by Value and Call by reference	C++ supports both call by value and call by reference.	Java supports call by value only. There is no call by reference in java.
Structure and Union	C++ supports structures and unions.	Java doesn't support structures and unions.
Thread Support	C++ doesn't have built-in support for threads. It relies on third-party libraries for thread support.	Java has built-in thread support.
Documentation comment	C++ doesn't support documentation comment.	Java supports documentation comment (/** ... */) to create documentation for java source code.

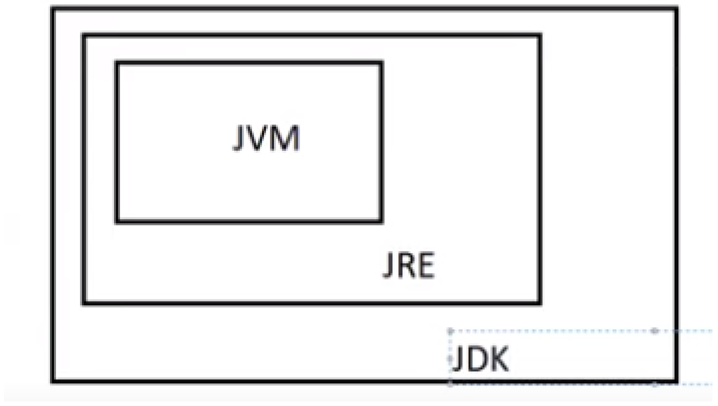
Contd…



Virtual Keyword	C++ supports virtual keyword so that we can decide whether or not override a function.	Java has no virtual keyword. We can override all non-static methods by default. In other words, non-static methods are virtual by default.
unsigned right shift >>>	C++ doesn't support >>> operator.	Java supports unsigned right shift >>> operator that fills zero at the top for the negative numbers. For positive numbers, it works same like >> operator.
Inheritance Tree	C++ creates a new inheritance tree always.	Java uses a single inheritance tree always because all classes are the child of Object class in java. The object class is the root of the inheritance tree in java.
Hardware	C++ is nearer to hardware.	Java is not so interactive with hardware.
Object-oriented	C++ is an object-oriented language. However, in C language, single root hierarchy is not possible.	Java is also an object-oriented language. However, everything (except fundamental types) is an object in Java. It is a single root hierarchy as everything gets derived from java.lang.Object.

Components Of Java Programming Language

- Java Development kit (JDK)
- Java Virtual Machine (JVM):
- Java Runtime Environment (JRE)



Java Development kit (JDK)

- JDK is a software development environment used for making applets and Java applications.
- JDK helps Java developers to code and run Java programs.
- JDK contains tools required to write/develop the Java programs and JRE to execute them.
  - It includes a compiler, Java application launcher, Appletviewer, etc.
  - Compiler converts code written in Java into byte code.
  - Java application launcher opens a JRE, loads the necessary class, and executes its main method.

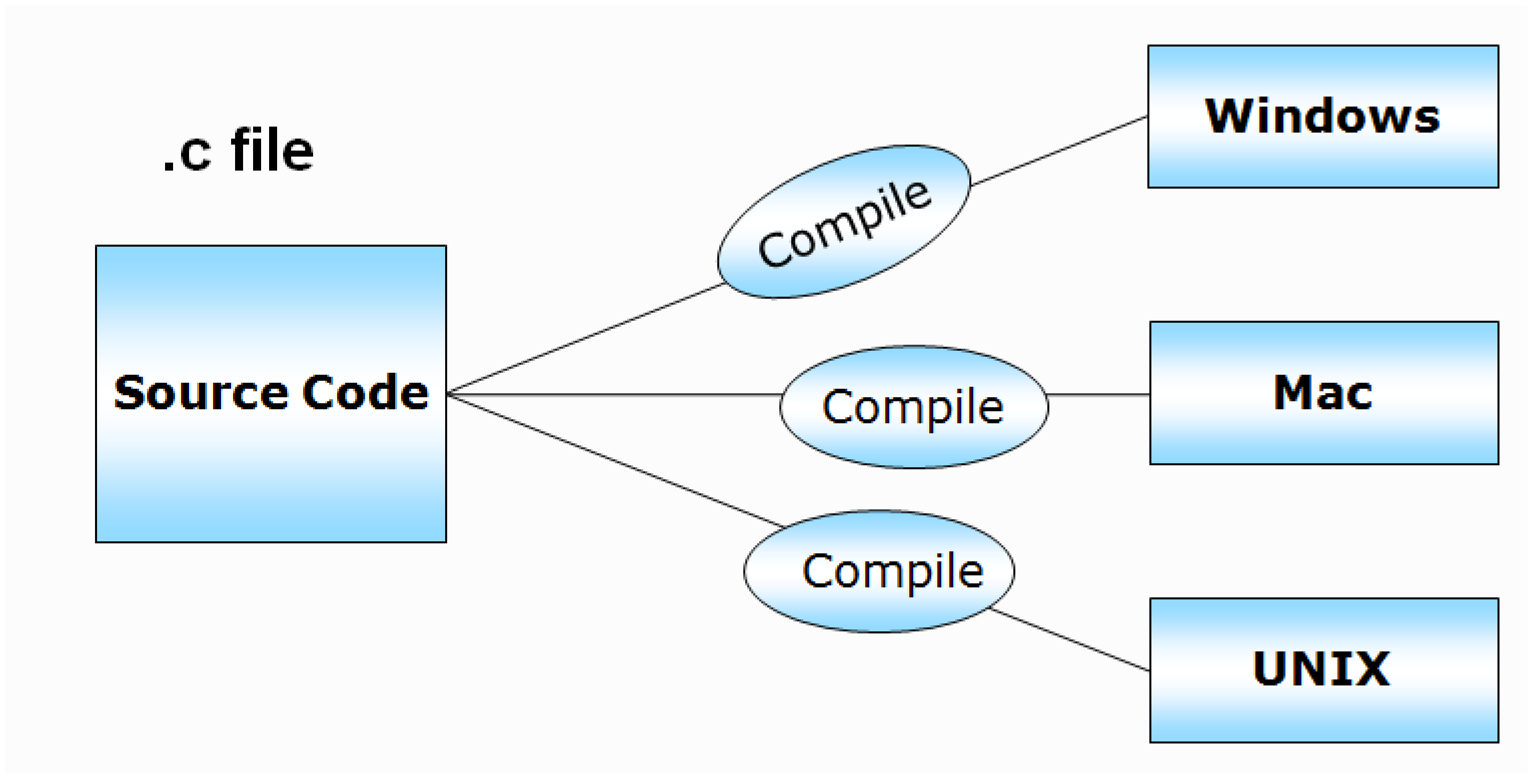
Java Runtime Environment (JRE)

- JRE is a piece of software that is designed to run other software.
- It contains the class libraries, loader class, and JVM.
- to run Java programs JRE must be installed .

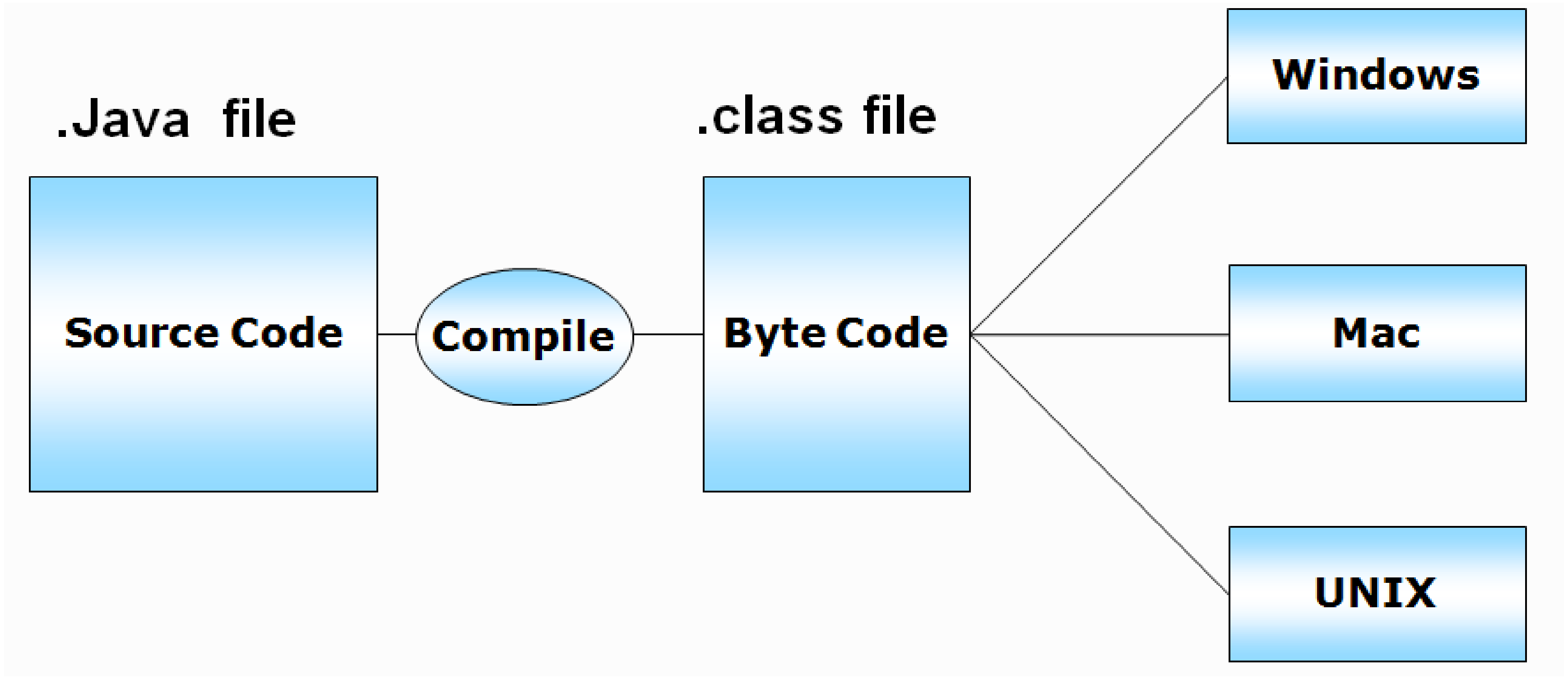
Java Virtual Machine (JVM)

- Java Virtual Machine (JVM) is an engine that provides a runtime environment to for Java Code or applications.
- It converts Java bytecode into machine language.
- JVM is a part of the Java Run Environment (JRE).
- In other programming languages, the compiler produces machine code for a particular system.
- But the Java compiler produces code for a Virtual Machine known as Java Virtual Machine.
- JVM is platform dependent.

C program Execution

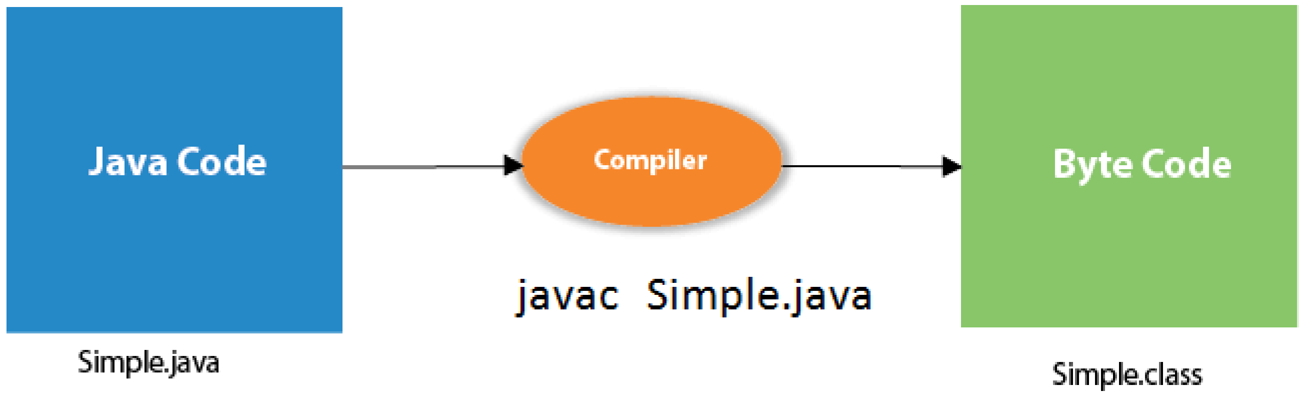


Java Program Execution

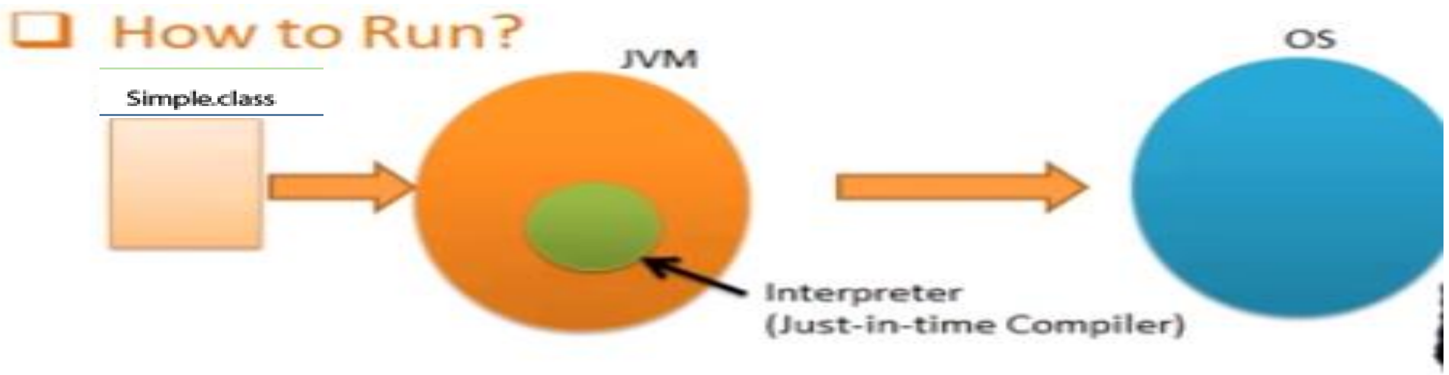


Contd…

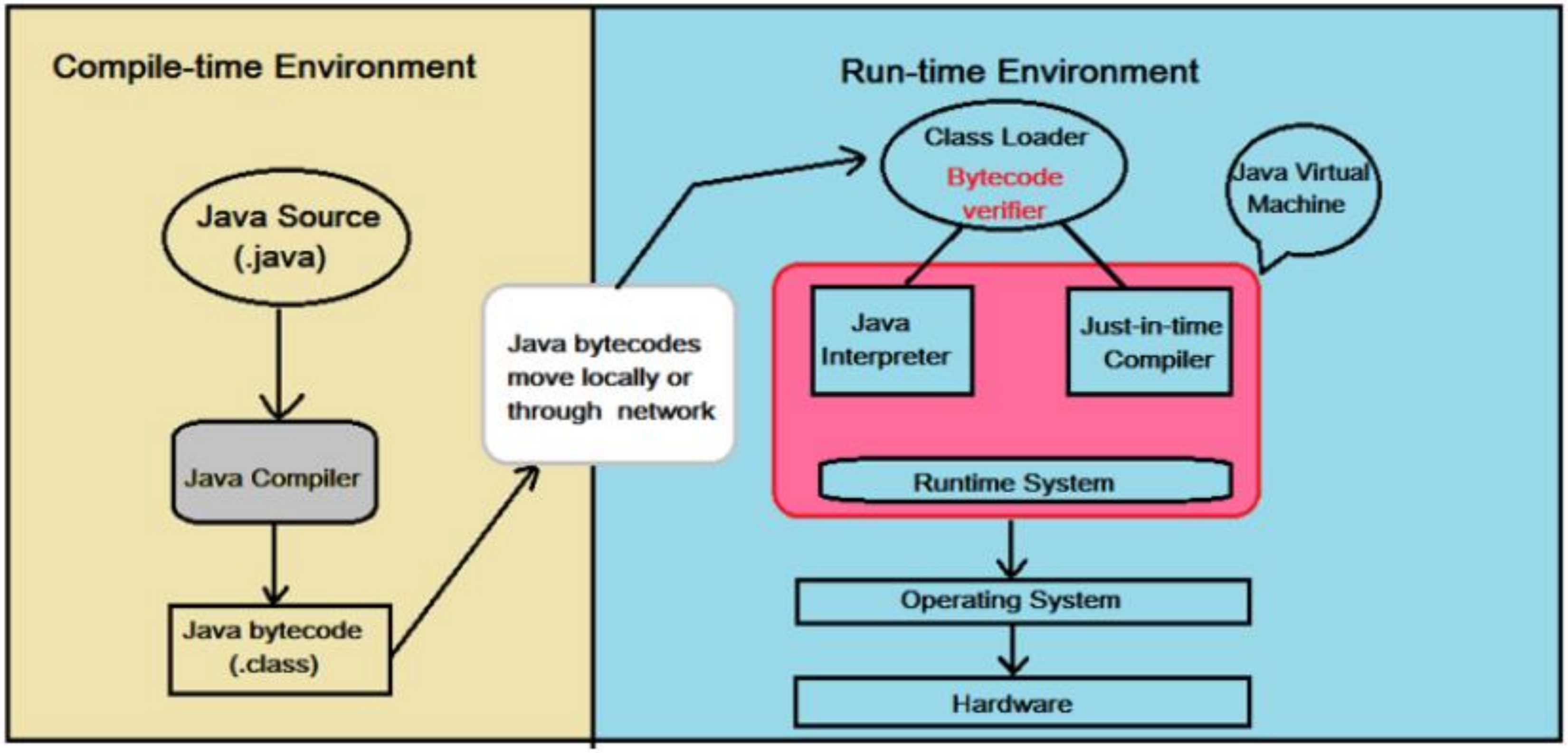
- What happens at compile time?

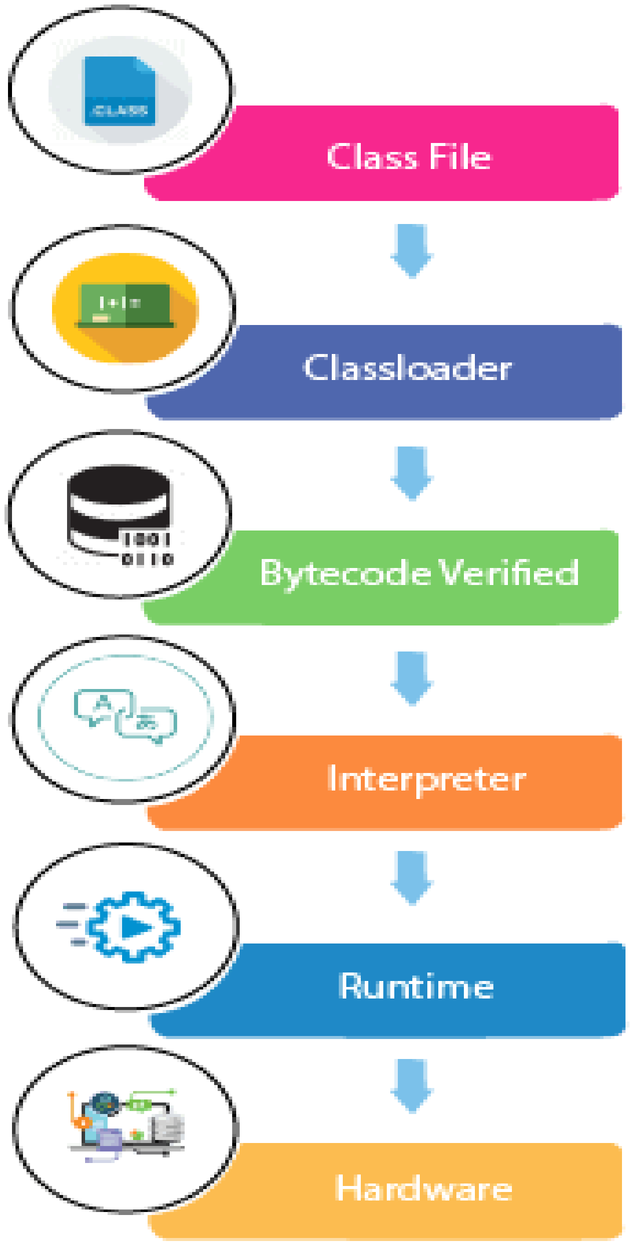


- What happens at runtime? (java simple)



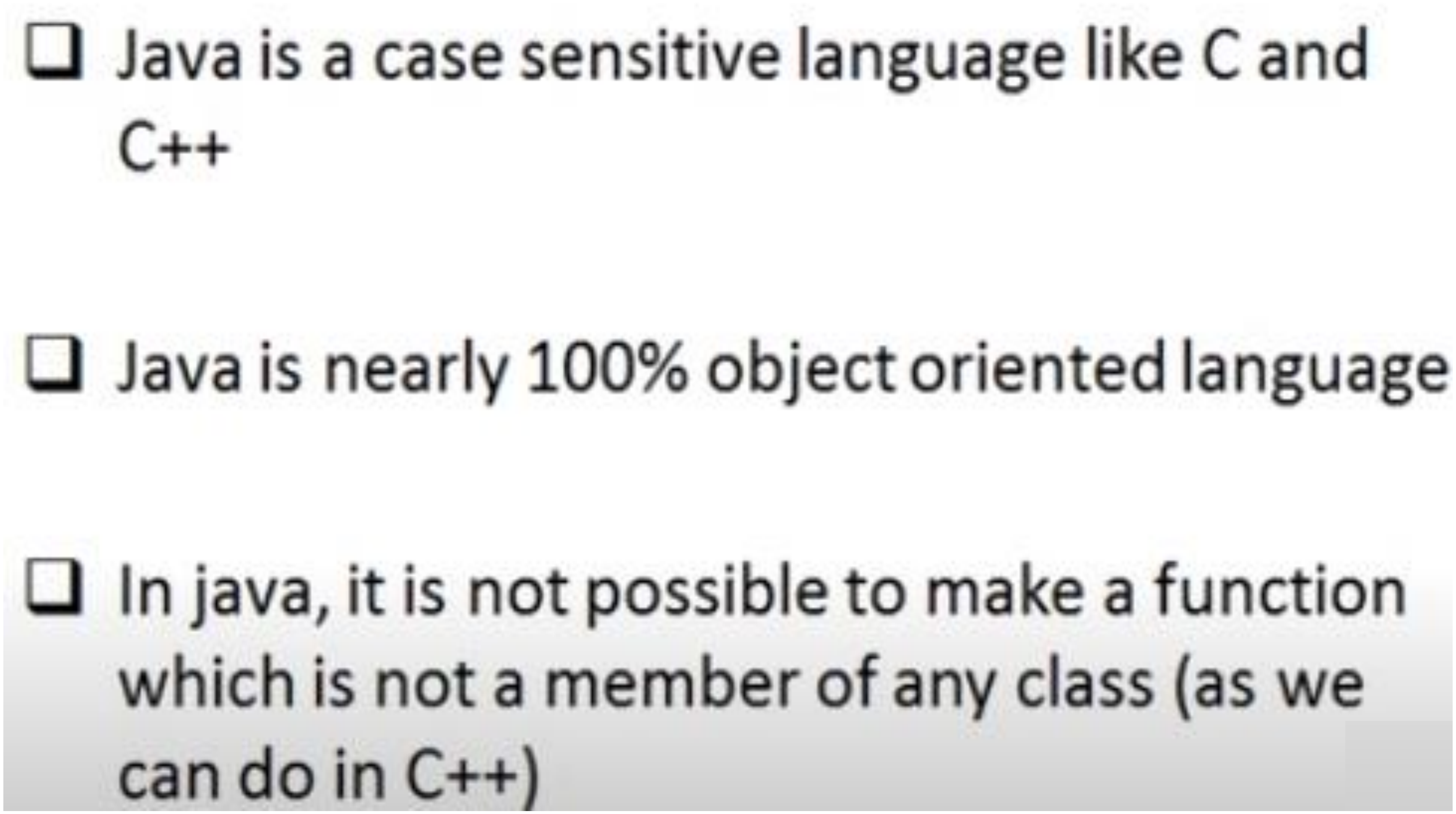
Contd…


Compile-time Environment
Java Source (.java)
Java Compiler
Java bytecode (.class)
Java bytecodes move locally or through network
Run-time Environment
Class Loader
Bytecode verifier
Java Virtual Machine
Java Interpreter
Just-in-time Compiler
Runtime System
Operating System
Hardware



- At runtime, following steps are performed:
- **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 objects.
- **Interpreter:** read bytecode stream then execute the instructions.

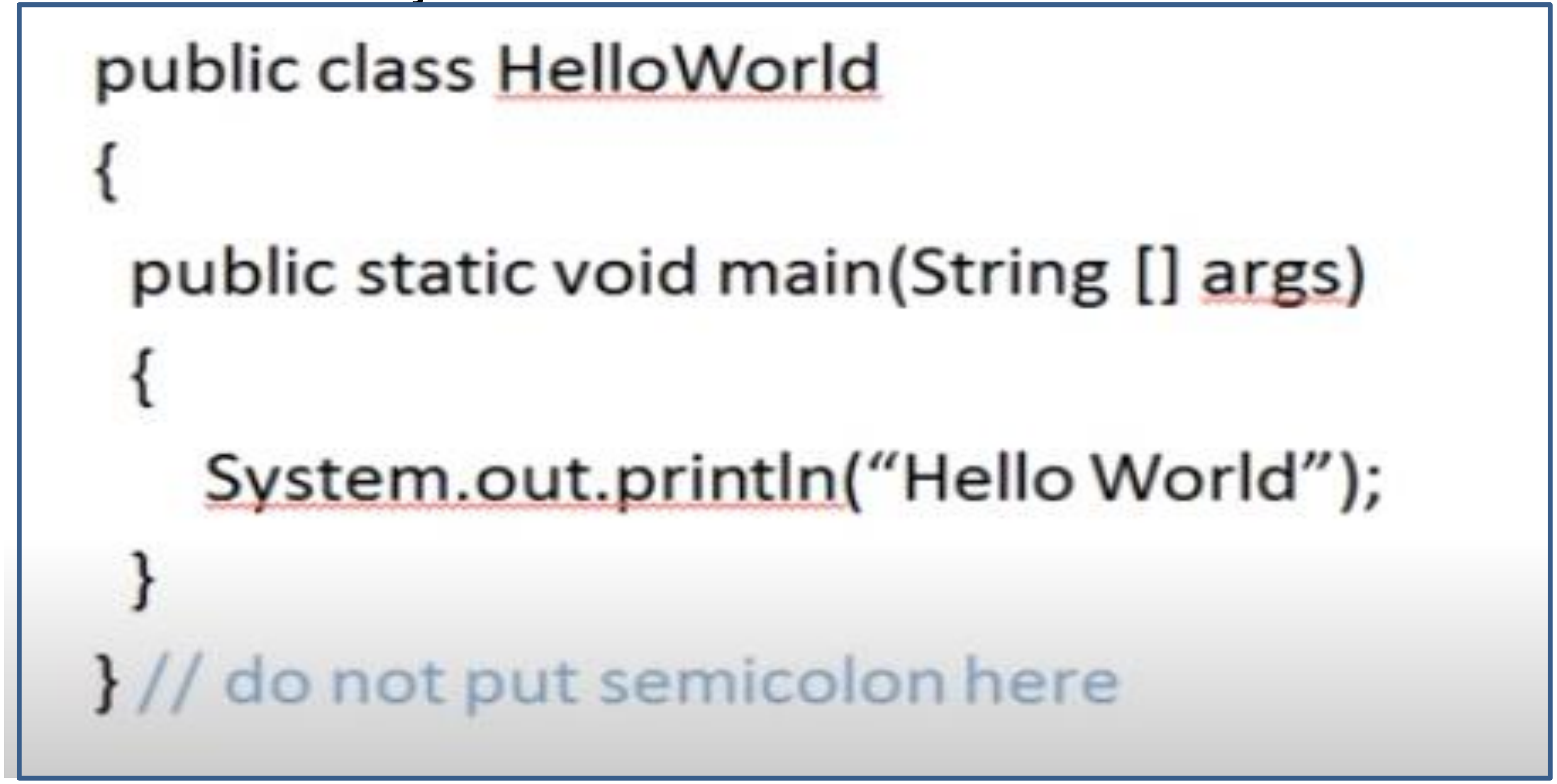
First program in Java



- Java is a case sensitive language like C and C++
- Java is nearly 100% object oriented language
- In java, it is not possible to make a function which is not a member of any class (as we can do in C++)

Creating Hello World Example

HelloWorld.java



```
public class HelloWorld
{
 public static void main(String [] args)
 {
    System.out.println("Hello World");
 }
} // do not put semicolon here
```

Parameters used in Java Program

- **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.
  - 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.

Contd…

- **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.