

# Introduction to Java

## 1.What is Java?

Java is a programming language used to make software, websites, mobile apps, and big systems.

## 2.Why Do We Need Java?

We need Java because:

- It works on all computers (Windows, Mac, Linux)
- It is fast and secure
- Big companies use it
- It is good for large programs
- java is portable.

## 3.What Problems Does Java Solve?

Java helps to:

- Build big applications (bank apps, websites, servers)
- Handle millions of users
- Keep data safe
- Run the same code on many devices

## 4.Platform Dependent Language

A platform dependent language runs only on one system (like only Windows or only Linux).

**Example:** C, C++ (needs recompilation for each system)

## 5.Platform Independent Language

A platform independent language runs on any system without changing the code.

**Example:** Java (Write Once, Run Anywhere)

- Platform dependent → Works on one platform
- Platform independent → Works on all platforms

## 6.Why C++ is Platform Dependent

1. C++ compiler converts code **directly into machine code**.
2. Machine code is **different for every CPU and OS** (Windows, Linux, Mac).
3. So the program works only on that system.

If you change system → you must recompile again.

## 7. Why Java is Platform Independent

1. Java compiler converts code into **Bytecode** (not machine code).
2. Bytecode runs on **JVM (Java Virtual Machine)**.
3. Every OS has its own JVM.
4. JVM converts bytecode to machine code automatically.

Same Java program runs everywhere.

But jvm is platform dependent

Bytecode is independent

## 8. Internal Flow

### C++

C++ Code → Machine Code → CPU

(Direct hardware dependent)

### Java

Java Code → Bytecode → JVM → Machine Code → CPU

(JVM makes it independent)

## Binary Code

Binary code is any code made of **0 and 1**.

Computers understand only 0 and 1.

## Machine Code

Machine code is **binary instructions made for a specific processor (CPU)**.

Directly runs on hardware.

# Bytecode

Bytecode is **intermediate code made by Java**, not for CPU.  
It runs on **JVM**, not directly on hardware.

## Problems in C / C++

C and C++ have these problems:

- Not platform independent
- Manual memory management (hard, errors)
- Less secure
- Complex for large systems
- No automatic garbage collection
- pointers
- Multiple Inheritance

## How Java Solves These Problems

Java solves them by:

- Platform independent (runs everywhere)
- Automatic memory management (Garbage Collector)
- More secure
- Easy for large applications
- Built-in libraries and OOP
- remove pointers and mutiple inheitage

## 9.How Java is PORTABLE (Write Once, Run Anywhere)

### Problem in C/C++

- C/C++ code becomes **machine code for one processor + OS**
- If you change Windows → Linux, you must recompile

### 1.How Java Solves This (Internally)

#### Step 1

Java code is converted to **Bytecode**

Java Code → Bytecode

## Step 2

Bytecode runs on **JVM (Java Virtual Machine)**

## Step 3

Every OS and processor has its own JVM

- Windows JVM
- Linux JVM
- macOS JVM

JVM converts bytecode to machine code automatically.

## Result

**Same Java program runs on Windows, Linux, Mac, Android.**

That's why Java is **platform independent and portable**.

# 2.How Java is SECURE

Java is secure because of **4 main layers**:

## 1. No Direct Memory Access

In C/C++ you can use pointers to change memory (dangerous).

Java **does not allow pointer manipulation**.

Hackers cannot easily access memory.

## 2. Bytecode Verification

Before running, JVM **checks bytecode**:

- No illegal code
- No memory corruption
- No virus-like instructions

If code is unsafe → JVM stops it.

## 3. Class Loader Security

Class Loader controls:

- Which class can be loaded

- From where (internet, disk)
- Prevents fake or harmful classes

## 4. Security Manager / Sandbox

Java programs run in **sandbox environment**:

- Cannot access system files without permission
- Cannot harm OS
- Safe for browsers, servers, apps

### Portability

**Java Code → Bytecode → JVM → Any OS/CPU**

### Security

**Code → Bytecode Check → Class Loader → Sandbox → Run Safely**