Welcome to my channel

Code With Amit

Feature of java

Java is a powerful language with many useful features.

It keeps improving with every new version.

Today, Java is used on billions of devices all over the world.

The main features of Java are:

1. Object-Oriented
2. Platform Independent
3. Simple
4. Secure
5. Architecture Neutral
6. Portable
7. Robust
8. Multithreaded
9. Interpreted
10. High Performance
11. Distributed
12. Dynamic

1. Object-Oriented

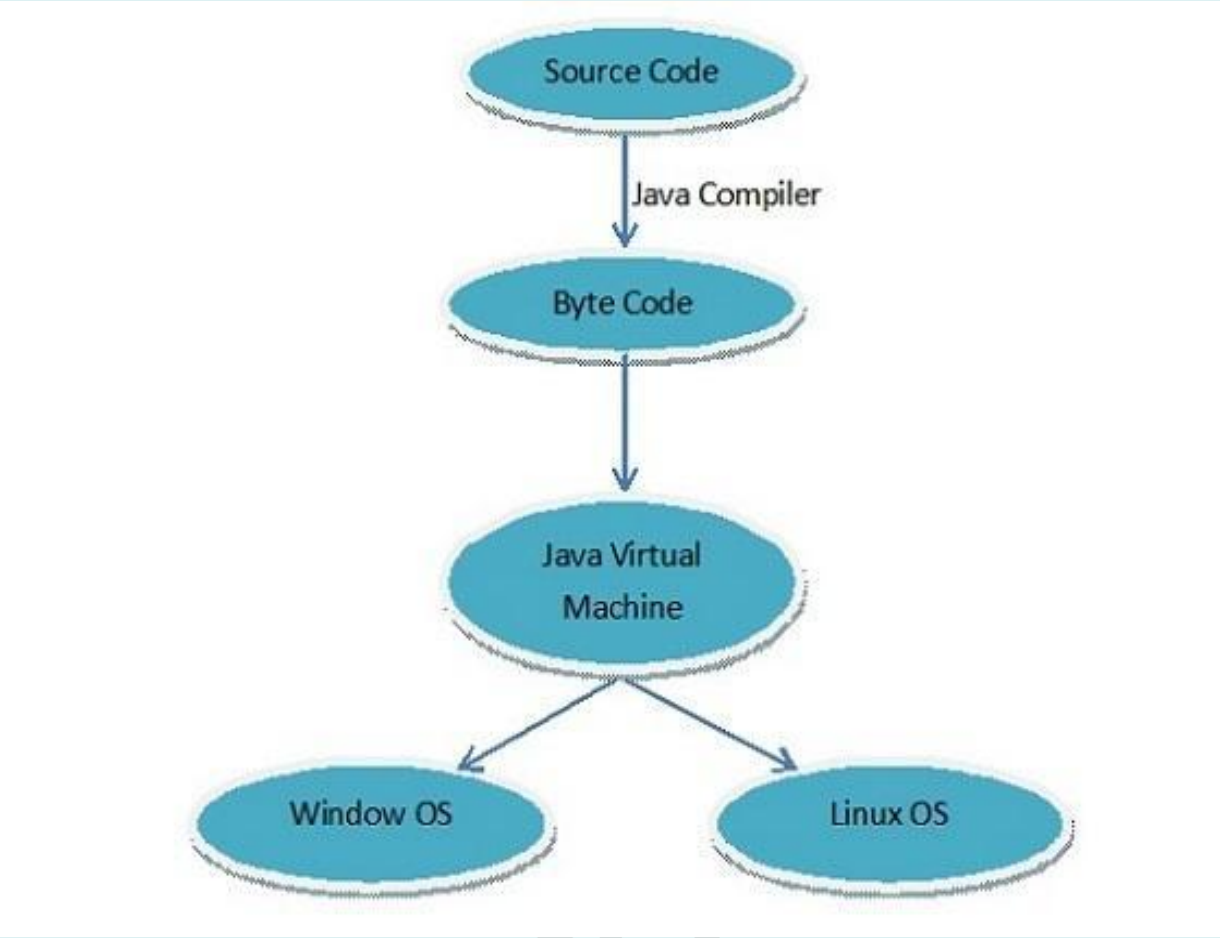
- **Meaning**:
  Java is an **Object-Oriented Programming (OOP) language**.
  This means everything in Java is written **inside a class**, and objects are created from those classes.

- **Key Idea**:
  - A **class** is like a **blueprint** (plan).
  - An **object** is like a **real thing** made from that blueprint.

- **In Java**:
  Even the `main ()` method is inside a class.
  We cannot write code outside a class.

- Think of a **class** as a **Car Design** (blueprint).
- Using that design, you can make many **Car Objects** (like BMW, Audi, Honda).

2. Platform Independent



Java is **Platform Independent.**
This means once We write Java code on one computer and we can run it on **any other computer or operating system** (Windows, Linux, Mac, etc.) without changing the code.

Java code is first converted into **bytecode** (a special form).

This bytecode can run on any system that has the **Java Virtual Machine (JVM)**.

JVM understands bytecode and runs it.

Java code (bytecode) can be run anywhere as long as JVM is there.

3. Simple is simple

Reasons why Java is Simple:

1. **No Header Files**
   - In C++, we must include header files (`#include <iostream>`).
   - In Java, no such thing. We just start coding inside a class.
2. **No Pointers**
   - Pointers (memory addresses) in C++ are very hard and unsafe.
   - Java removed direct pointer use.
   - It only uses simple **references** for objects → safe and easy.
3. **No Operator Overloading**
   - In C++, you can give new meanings to operators (like + or -).
   - This makes code confusing.
   - Java keeps it simple → + is only for numbers addition or string joining.

4. **Automatic Memory Management (Garbage Collection)**
   - In C++, you must free memory manually (delete).
   - In Java, memory is managed automatically by the **Garbage Collector.**
   - This reduces errors and makes coding simple.
5. **Simple Syntax**
   - Java syntax looks like C/C++, so it feels familiar.
   - But it removed complicated parts (like multiple inheritance with classes).

4. Java is Secure?

1. **No Pointers**
   - In C/C++, **pointers** can directly access memory.
   - **Hackers** can **misuse** this to steal or damage data.
   - **Java removed pointers** → making memory safe.

2. **Bytecode + JVM**
   - Java code is first converted into **bytecode.**
   - This bytecode runs inside the **Java Virtual Machine (JVM).**
   - JVM checks the code before running it (called **Bytecode Verification**) to prevent harmful code.

3. **Automatic Memory Management**
   - Java has **Garbage Collection** that clears unused memory.
   - This avoids problems like **memory leaks** and makes programs safe.

4. **Access Control**
   - Java provides **access modifiers** (**private, protected, public**) to control who can use data and methods.

5. Architecture Neutral

Java is **architecture neutral**, which means **Java code does not depend on the type of computer** (hardware) or operating system (Windows, Linux, Mac).

Once compiled, Java code becomes **bytecode.**

This bytecode can run on **any system that has JVM (Java Virtual Machine).**

Java code → compiled into **bytecode.**

Bytecode is **independent of computer architecture** and needs only JVM to run anywhere.

6. Portable

Java is **portable,** which means we can write Java code on one system and run it on another system **without any changes.**

7. Robust

- ⇨ **No pointers** (safe memory).
- ⇨ **Automatic memory management.**
- ⇨ **Strong error handling** with try-catch.
- ⇨ **Strict type** checking.

8. Multithreaded

Multithreading means a program can do **many tasks at the same time.** Each task runs in a **separate thread** (small unit of execution).

Java supports multithreading **built-in.**

It makes programs faster and more efficient.

9. Interpreted

Java supports cross-platform code though the use of java bytecode.

Bytecode can be interpreted on any platform by the JVM.

10. High Performance

Java gives **better performance** compared to normal interpreted languages because it uses a special feature called **JIT (Just-In-Time) Compiler**.

Normally, interpreted languages are slower because they execute line by line.

In Java:

Code → compiled into **bytecode**.

JVM uses **JIT Compiler** to convert bytecode into **machine code** (native code) at run-time.

Machine code runs **directly on the CPU,** making it much faster.

11. Distributed

- Java is called **distributed** because it allows programs to be shared and run on different computers connected through a **network**.

- Java makes it easy to build applications that **communicate with each other** over the internet.

- Java has built-in support for networking (in the java.net package).

12. Dynamic

- Java can **load classes, methods, and objects at runtime** when needed.
- It also stores a lot of **run-time information** about objects.

Thankyou!