JAVA v.s. C++
Programming Language Comparison

By

LI LU

SAMMY CHU

Object-oriented Programming Language

- ***Both Java and C++ are most popular object-oriented programming languages***
- ***C++ was created at AT&T Bell Labs in 1979***
- ***Java was born in Sun Microsystems in 1990***

Language Feature Comparison

- *Simple*
- *Object-oriented*
- *Distributed*
- *Robust*
- *Secure*
- *Architecture Neutral*
- *Portable*
- *Compiled or Interpreted*
- *High Performance*
- *Multithreaded*
- *Dynamic*
- *Fun*

Simple

JAVA	C++
No pointer	Pointer
No multiple inheritance	Multiple inheritance
Automatic garbage collection	Manual garbage collection
No operator overloading	Operator overloading
No goto statement and no structure and union data structure	Goto statement and structure and union data structure

Purely Object-oriented

JAVA

- *No stand-alone data and functions*
- *Automatically supports polymorphism*

Hybrid Object-oriented

C++

- *Allows the stand-alone data and functions*
- *Needs declare virtual methods explicitly*

Distributed

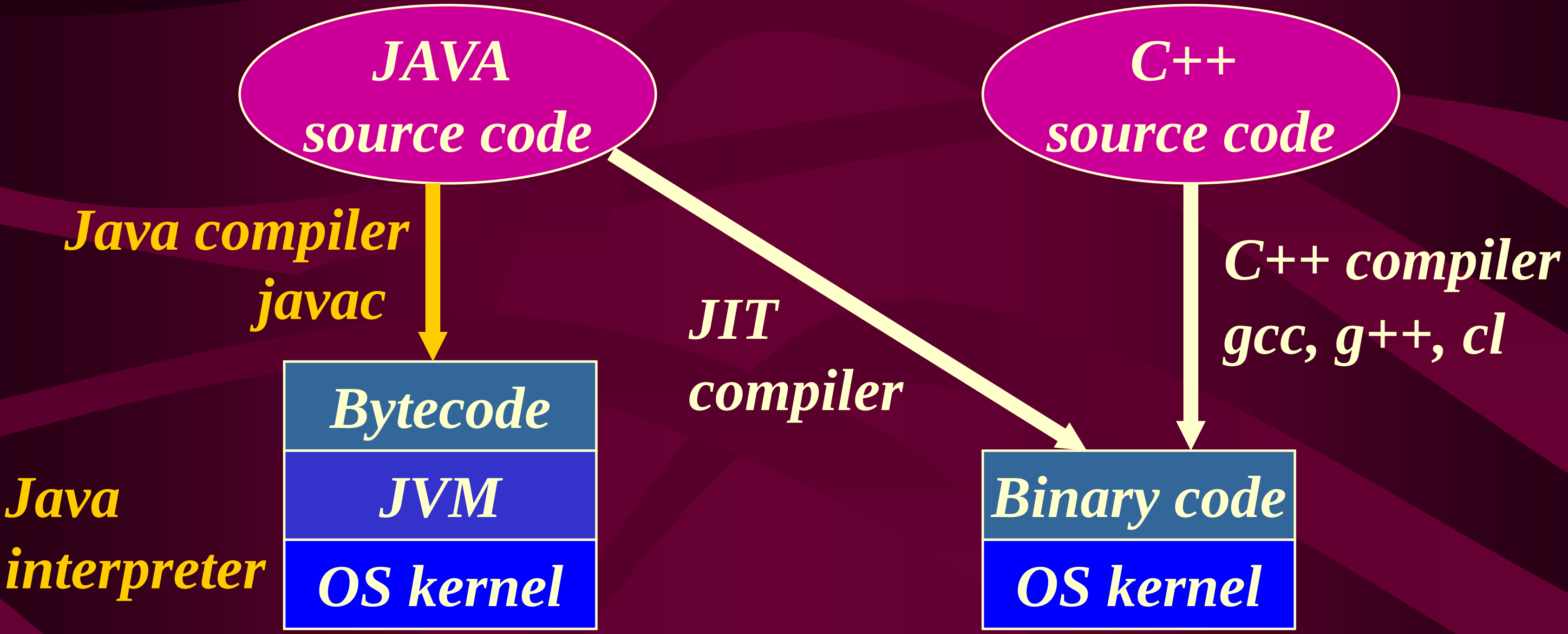
JAVA

Handles TCP/IP networking easily and nicely, can open and access objects across the Internet via URL just like a local file system

C++

External library supports TCP/IP networking, but much harder to do network programming

Interpreted or Compiled



JAVA source code

Java compiler javac

Bytecode

JVM

OS kernel

Java interpreter

JIT compiler

C++ source code

C++ compiler gcc, g++, cl

Binary code

OS kernel

Both compiled and interpreted

Compiled

High Performance

JAVA

- *Much slower than C++, but good enough to run interactively for most applications*
- *JIT compiler available*

C++

- *About 10~20 times faster than equivalent JAVA code*
- *Most operating systems are written using C/C++*

Robust

JAVA

- *Originally designed for writing highly reliable or robust software*
- *Explicit method declarations*
- *No pointers and automatic garbage collection avoid hard-to-debug mistakes*
- *Array bounds-checking*

C++

- *Allows implicit type and function declarations*
- *No automatic garbage collection is susceptible to memory leakage*
- *Using pointers is susceptible to memory corruption*
- *No array bounds-checking*

Secure

JAVA

- *Byte-code is verified at run-time to ensure security restrictions are not violated*
- *Memory layout is handled at run-time by JVM*
- *Uses multiple namespaces to prevent hostile classes from spoofing a JAVA program*

C++

- *Memory is handled at compile-time by compiler*

Architecture neutral and Portable

JAVA

- *Same Bytecode can run on any machine supporting JVM*
- *Well defined and fixed-size data types, file formats, and GUI behavior*

C++

- *Platform-dependent binary code cannot be executed on a different machine*
- *Implementation specific and varied-size data types by platforms*

Multithreaded

JAVA

- *Provides native multithreading support*
- *Concurrent applications are quite easy*

C++

- *Rely on external libraries for multithreading*
- *Harder to do multithreaded programming*

Dynamic

JAVA

- *Run-time representation for classes makes it possible to dynamically link classes into a running system*
- *Loads classes as needed, even from across networks*

C++

- *Needs recompile if libraries are updated*
- *Load libraries when compiled*

FUN FUN FUN

JAVA *Nice features combined with the Internet applications make JAVA programming appealing and fun*

C++ *The complicated or even some confusing features make C++ programming error prone*

Summary and Conclusion

- *C++ is a high performance and powerful language. Most of the industry software is written in C/C++*

- *JAVA's cross-platform compatibility and convenient APIs for networking and multi-threading have won it a place in the business world. Java is the logically next step in the evolution of C++*