

C++ vs Java

| | |
|-----------|----------------------------|
| 📅 Created | @January 21, 2025 11:35 PM |
| 📄 Class | Prog2 |

C++: Linguaggio compilato

C++ è un linguaggio **compilato**, il che significa che il codice sorgente viene tradotto direttamente in codice macchina (binario) attraverso un compilatore, come GCC o Clang. Questo codice macchina è specifico per il sistema operativo e l'architettura hardware del computer su cui è stato compilato.

1. Processo di esecuzione:

- Il programmatore scrive il codice sorgente (.cpp).
- Il compilatore converte il codice sorgente in un file eseguibile nativo (es., `.exe` su Windows o un binario su Linux).
- Il file eseguibile può essere eseguito direttamente dalla CPU senza ulteriori traduzioni.

2. Caratteristiche principali:

- **Performance:** Essendo direttamente tradotto in codice macchina, il codice C++ tende a essere più veloce rispetto ai linguaggi interpretati o semi-interpretati.
- **Portabilità limitata:** Il file eseguibile generato è specifico per l'architettura e il sistema operativo, quindi non può essere eseguito su piattaforme diverse senza ricompilare il codice sorgente.
- **Accesso hardware:** C++ permette un accesso più diretto alle risorse hardware, grazie alla sua vicinanza al linguaggio macchina.

Java: Linguaggio semi-interpretato (bytecode e JVM)

Java è considerato un linguaggio **semi-interpretato**. Il codice sorgente Java viene prima compilato in **bytecode**, un linguaggio intermedio. Questo bytecode non può

essere eseguito direttamente dall'hardware ma richiede una macchina virtuale Java (**JVM**) per essere interpretato o ulteriormente compilato.

1. Processo di esecuzione:

- Il programmatore scrive il codice sorgente (.java).
- Il compilatore Java (`javac`) traduce il codice sorgente in bytecode (.class), un formato portabile e indipendente dalla piattaforma.
- Il bytecode viene eseguito dalla JVM:
 - In modalità interpretativa: La JVM traduce il bytecode in istruzioni macchina al volo.
 - In modalità JIT (Just-In-Time compilation): Parti del bytecode vengono compilate in codice nativo durante l'esecuzione per migliorare le prestazioni.

2. Caratteristiche principali:

- **Portabilità:** Il bytecode è indipendente dalla piattaforma, quindi lo stesso programma Java può essere eseguito su qualsiasi sistema che abbia una JVM compatibile.
- **Performance:** Sebbene inizialmente più lenta rispetto a C++ (per via della traduzione al runtime), l'uso di JIT consente a Java di avvicinarsi alle prestazioni di un linguaggio compilato.
- **Sicurezza:** Java opera in un ambiente virtuale (JVM), limitando l'accesso diretto all'hardware e riducendo il rischio di comportamenti imprevisti o dannosi.

Differenze principali tra C++ e Java

| Aspetto | C++ (Compilato) | Java (Interpretato) |
|------------------------|------------------------------|------------------------------------|
| Output del compilatore | File eseguibile nativo | Bytecode (.class) |
| Esecuzione | Direttamente dalla CPU | Tramite JVM |
| Portabilità | Dipendente dall'architettura | Altamente portabile (richiede JVM) |

| | | |
|-----------------------------|---|--|
| Prestazioni | Generalmente più veloce (codice nativo) | Più lento inizialmente, migliorato con JIT |
| Accesso hardware | Accesso diretto alle risorse | Limitato dalla sandbox della JVM |
| Compilazione runtime | Non applicabile | JIT durante l'esecuzione |

Conclusione

- **C++** è ideale per applicazioni che richiedono alte prestazioni e un controllo preciso sull'hardware (come sistemi operativi, giochi, e applicazioni embedded).
- **Java** eccelle in portabilità, sicurezza e semplicità per applicazioni multiplatforma (come web application, strumenti aziendali e app mobili).

Entrambi hanno vantaggi e limitazioni specifiche e la scelta dipende dal contesto del progetto e dai requisiti dell'applicazione.