Introduzione al linguaggio Java

Docente
Mario Perna
prof.perna.mario@darzo.net

A.S.
2025/2026
Materia
Informatica

Introduzione

Negli anni precedenti siamo stati abituati a realizzare programmi scritti con il linguaggio di programmazione **C++.**

In questo anno scolastico impareremo il linguaggio di programmazione **Java** che condivide varie similitudini con il linguaggio **C++**, ma anche sostanziali differenze che analizzeremo più avanti.

Perché imparare **Java**?

- Top 10 dei linguaggi più popolari
- Usato a livello aziendale (framework Spring)
- Usato per realizzare giochi (Minecraft)
- Punto di partenza per lo sviluppo di app Android
- In passato molto diffuso anche per lo sviluppo web (ad oggi poco diffuso)
- Sintassi simile al **C++**

Caratteristiche del linguaggio Java

Java è un linguaggio di programmazione **orientato agli oggetti**, progettato per avere il minor numero possibile di dipendenze di implementazione.

È concepito per consentire agli sviluppatori di applicazioni di **scrivere una volta ed eseguire ovunque**, il che significa che il codice Java compilato può essere eseguito su tutte le piattaforme che supportano Java senza la necessità di ricompilazione.

Java è stato rilasciato per la prima volta nel 1995 ed è ampiamente utilizzato per lo sviluppo di applicazioni per dispositivi desktop, web e mobili. Java è noto per la sua semplicità, robustezza e funzionalità di sicurezza, il che lo rende una scelta popolare per le applicazioni di livello aziendale.

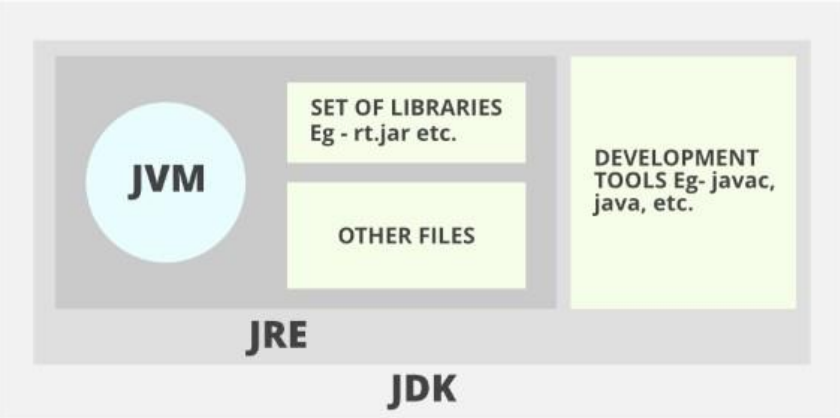
Componenti per lo sviluppo Java

Lo sviluppo di programmi in linguaggio Java è composto da 3 elementi fondamentali:

- **JVM (Java Virtual Machine):** componente software che esegue i programmi Java.

- **JRE (Java Runtime Environment):** consiste in un software che mette a disposizione l'ambiente per utilizzare la **JVM** (e infatti la contiene). Per potere eseguire codice Java bisogna scaricare il **JRE** (non è possibile scaricare solo la **JVM**).

- **JDK (Java Development Kit):** è il software minimo che serve per sviluppare in Java. Contiene il **JRE** (e quindi la **JVM**) e una suite di strumenti per sviluppare, tra cui il già citato compilatore.



Caratteristiche del linguaggio Java

- **Indipendente dalla piattaforma**

Il compilatore converte il **codice sorgente** in **bytecode** e poi la **JVM** esegue il **bytecode** generato dal compilatore. Questo **bytecode** può essere eseguito su qualsiasi piattaforma, che sia Windows, Linux o macOS, il che significa che se compiliamo un programma su Windows, possiamo eseguirlo su Linux e viceversa. Ogni sistema operativo ha una JVM diversa, ma l'output prodotto da tutti i sistemi operativi è lo stesso dopo l'esecuzione del **bytecode**. Ecco perché chiamiamo Java un linguaggio indipendente dalla piattaforma.

- **Programmazione orientata agli oggetti**

Java è un linguaggio orientato agli oggetti, che promuove l'uso di oggetti e classi. Il significato di questi due termini verrà approfondito più avanti. Per ora consideratela come un'estensione della programmazione imperativa.

Caratteristiche del linguaggio Java

- **Semplicità**

La sintassi di Java è semplice e facile da imparare, soprattutto per chi ha familiarità con C o C++. Elimina funzionalità complesse come puntatori e gestione esplicita della memoria rendendo più semplice scrivere, eseguire il debug e gestire il codice.

- **Robustezza**

Il linguaggio Java è robusto, ovvero affidabile. È sviluppato in modo tale da impegnarsi molto nel controllo degli errori il prima possibile, ecco perché il compilatore Java è in grado di rilevare anche quegli errori che non sono facili da rilevare con un altro linguaggio di programmazione. Le caratteristiche principali di Java che lo rendono robusto sono la **garbage collection**, la **gestione delle eccezioni** e **l'allocazione della memoria**.

Caratteristiche del linguaggio Java

- **Sicurezza**

In Java, non abbiamo puntatori, quindi non possiamo accedere ad array fuori limite. Ecco perché diversi difetti di sicurezza come la corruzione dello stack o il buffer overflow sono impossibili da sfruttare in Java. Inoltre, i programmi Java vengono eseguiti in un ambiente indipendente dall'ambiente del sistema operativo, il che rende i programmi Java più sicuri.

- **Meno performante di altri linguaggi**

Il linguaggio Java rispetto ad esempio al C++ deve assicurarsi tutta una serie di controlli che richiedono tempo e risorse. Ad esempio, in altri linguaggi è lasciato in carico al programmatore assicurarsi dell'utilizzo efficiente della memoria. Questo meccanismo provoca una degradazione delle performance seppur limitata dall'utilizzo di un linguaggio intermedio definito **bytecode**.

Come funziona un programma Java

1. **Scrittura del codice sorgente**

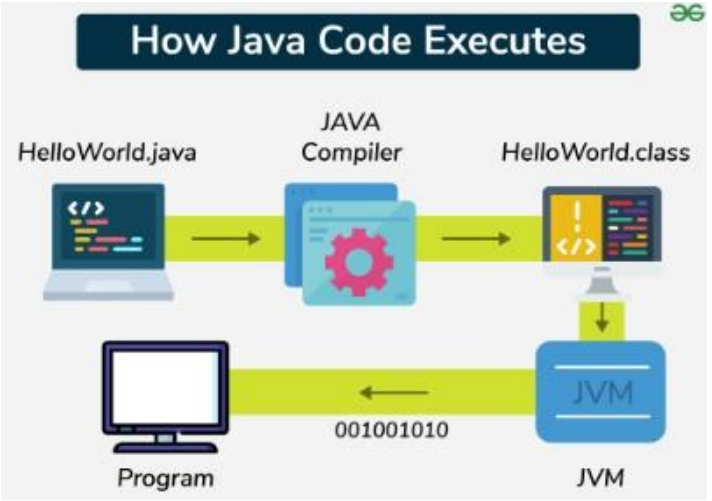
I programmi Java vengono scritti utilizzando un editor di testo o un ambiente di sviluppo integrato (IDE) come IntelliJ IDEA, Eclipse o NetBeans. Il codice sorgente viene salvato con un'estensione .java.

2. **Compilazione del programma**

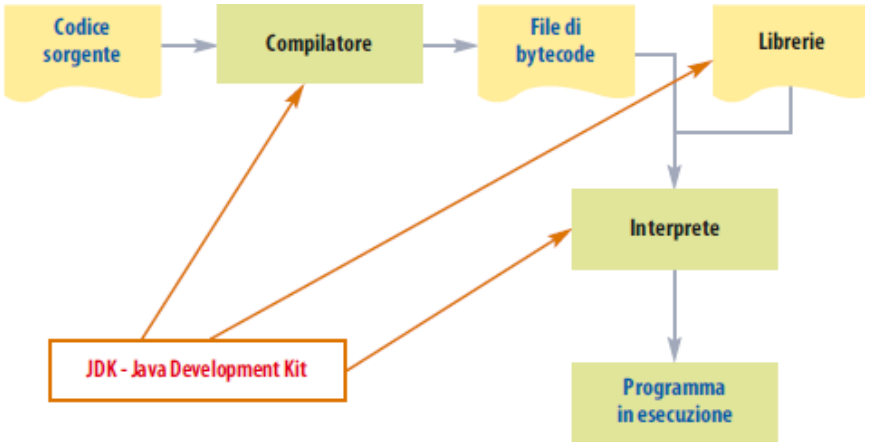
Il compilatore Java (javac) converte il codice sorgente in bytecode, che viene memorizzato in un file .class. Questo bytecode è indipendente dalla piattaforma e può essere eseguito su qualsiasi macchina con una JVM.

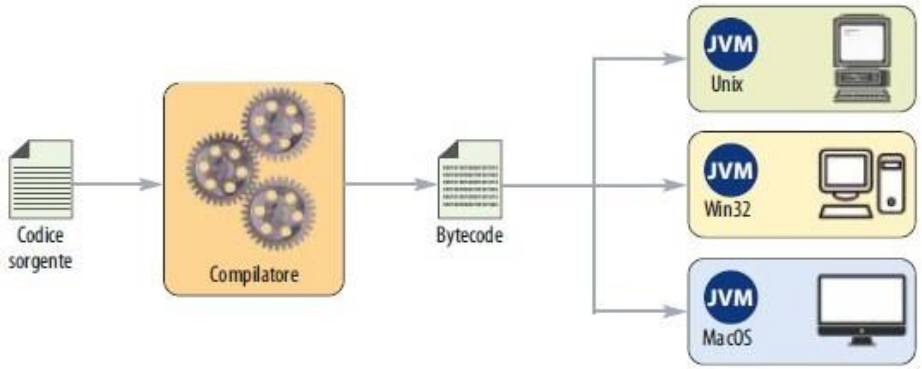
3. **Esecuzione del programma**

La JVM esegue il bytecode compilato, traducendolo in codice macchina specifico per il sistema operativo e l'hardware. In questa fase il bytecode viene interpretato.



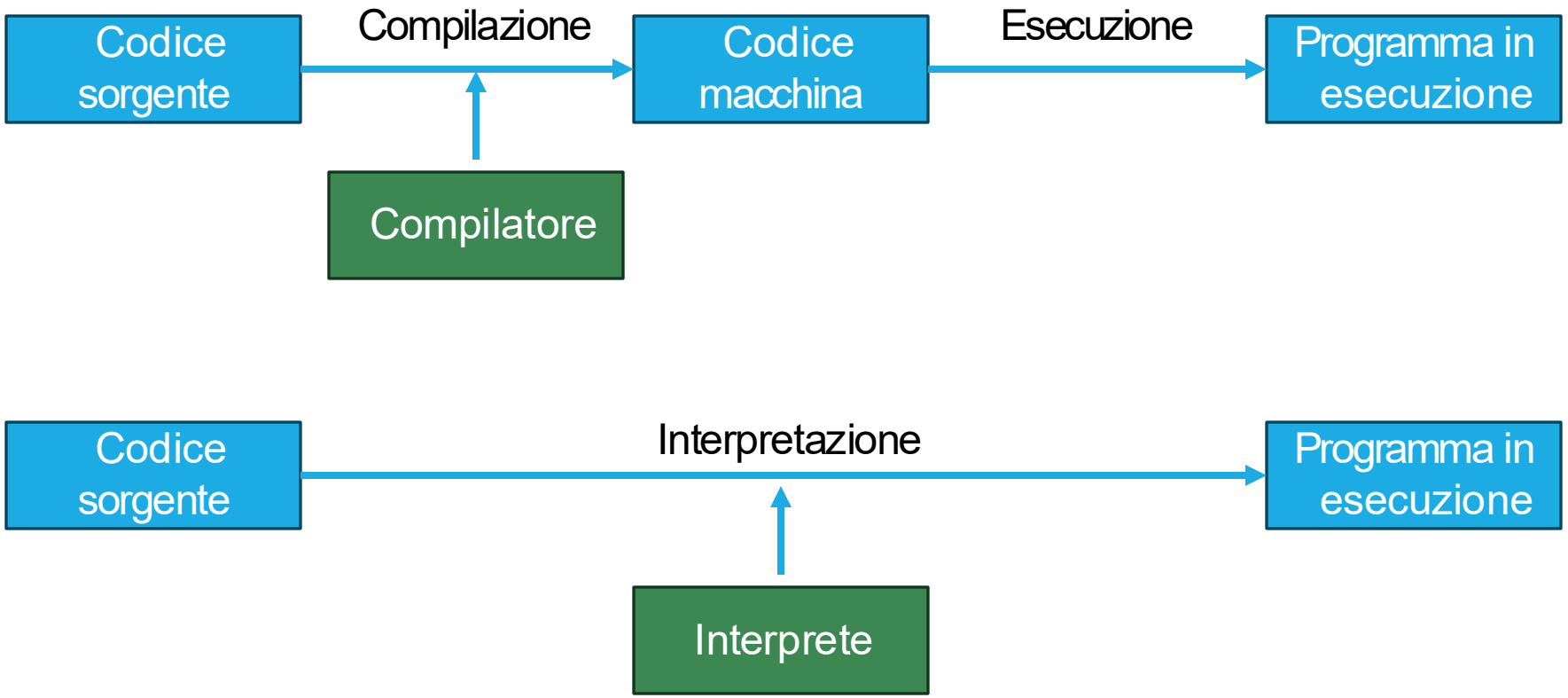
Come funziona un programma Java





Java è uno dei pochi linguaggi che è sia compilato che interpretato cercando di combinare i pregi delle due categorie.

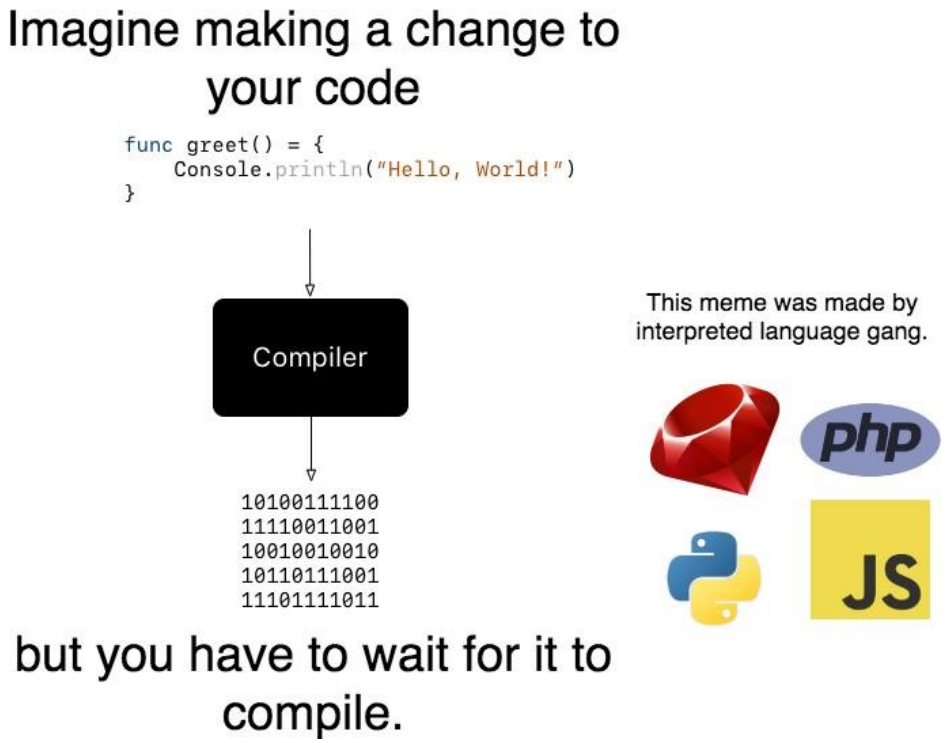
Differenze tra compilazione e interpretazione



Differenze tra compilazione e interpretazione

Linguaggio compilato	Linguaggio interpretato
Un linguaggio compilato è un linguaggio di programmazione le cui implementazioni sono in genere compilatori e non interpreti.	Un linguaggio interpretato è un linguaggio di programmazione le cui implementazioni eseguono le istruzioni direttamente, senza dover prima compilare un programma in istruzioni in linguaggio macchina.
Una volta compilato, il programma viene espresso nelle istruzioni della macchina di destinazione.	Le istruzioni non vengono eseguite direttamente dalla macchina di destinazione.
Ci sono almeno due passaggi per passare dal codice sorgente all'esecuzione.	Per passare dal codice sorgente all'esecuzione è necessario un solo passaggio.
I programmi compilati vengono eseguiti più velocemente di quelli interpretati.	I programmi interpretati possono essere modificati mentre sono in esecuzione.
Errori di compilazione impediscono la compilazione del codice.	Tutto il debug avviene in fase di esecuzione.
Java, C, C++, C#, COBOL, Rust, ecc.	**Java**, JavaScript, Perl, Python, ecc.

Differenze tra compilazione e interpretazione



Informazioni sull’utilizzo dei materiali didattici

Queste slides si basano su materiali originariamente elaborati dal **Prof. Andrea Melioli**, opportunamente modificati e integrati secondo specifiche esigenze riguardanti la programmazione disciplinare.

L’autore autorizza al **prof. Mario Perna** l’utilizzo, la modifica, la pubblicazione e qualsiasi altra forma di operazione sui materiali a scopo didattico e formativo.

L’uso o la diffusione di questi materiali è **vietato** senza il preventivo contatto con il proprietario del documento, Prof. Mario Perna (prof.mario.perna@darzo.net).