



# Introduzione a Java

*Programmazione*

**Java è un linguaggio di programmazione ad oggetti, fortemente tipizzato, compilato e multiplatforma, progettato per garantire portabilità, sicurezza e robustezza.**

**È stato sviluppato da Sun Microsystems nel 1995 (poi acquisita da Oracle Corporation) con l'obiettivo di creare applicazioni eseguibili su qualunque sistema tramite il principio:**

**“Write Once, Run Anywhere” (WORA)**



## Architettura tecnica

**Java non compila direttamente in codice macchina nativo, ma segue questo flusso:**

- 1. Codice sorgente (.java)**
- 2. Compilazione tramite javac**
- 3. Generazione di bytecode (.class)**
- 4. Esecuzione sulla Java Virtual Machine (JVM)**

**La JVM interpreta o compila Just-In-Time (JIT) il bytecode in codice macchina specifico per il sistema operativo.**



# **Caratteristiche principali**

## **Object-Oriented**

### **Basato su:**

- **Classi e oggetti**
- **Incapsulamento**
- **Ereditarietà**
- **Polimorfismo**



## Fortemente tipizzato

Ogni variabile deve avere un tipo dichiarato:

1. **int** numero = 10;
2. **String** nome = "Mirko";



## **Gestione automatica della memoria**

**Include Garbage Collector che libera automaticamente memoria non più utilizzata.**

## **Sicurezza**

- **Nessun accesso diretto alla memoria**
- **Controllo dei tipi a runtime**
- **Sandboxing nella JVM**



# Ambiti di utilizzo

**Java è utilizzato in:**

- **Applicazioni enterprise (es. Spring)**
- **Backend web**
- **Sistemi bancari e finanziari**
- **Android (tramite JVM-like runtime)**
- **Big Data (es. Apache Hadoop)**
- **Sistemi distribuiti**



# Ecosistema

## Java include:

- **JDK (Java Development Kit) → strumenti di sviluppo**
- **JRE (Java Runtime Environment) → ambiente di esecuzione**
- **Librerie standard molto estese**
- **Framework enterprise maturi**





**Java richiede il JDK (Java Development Kit).**

**Puoi scegliere tra:**

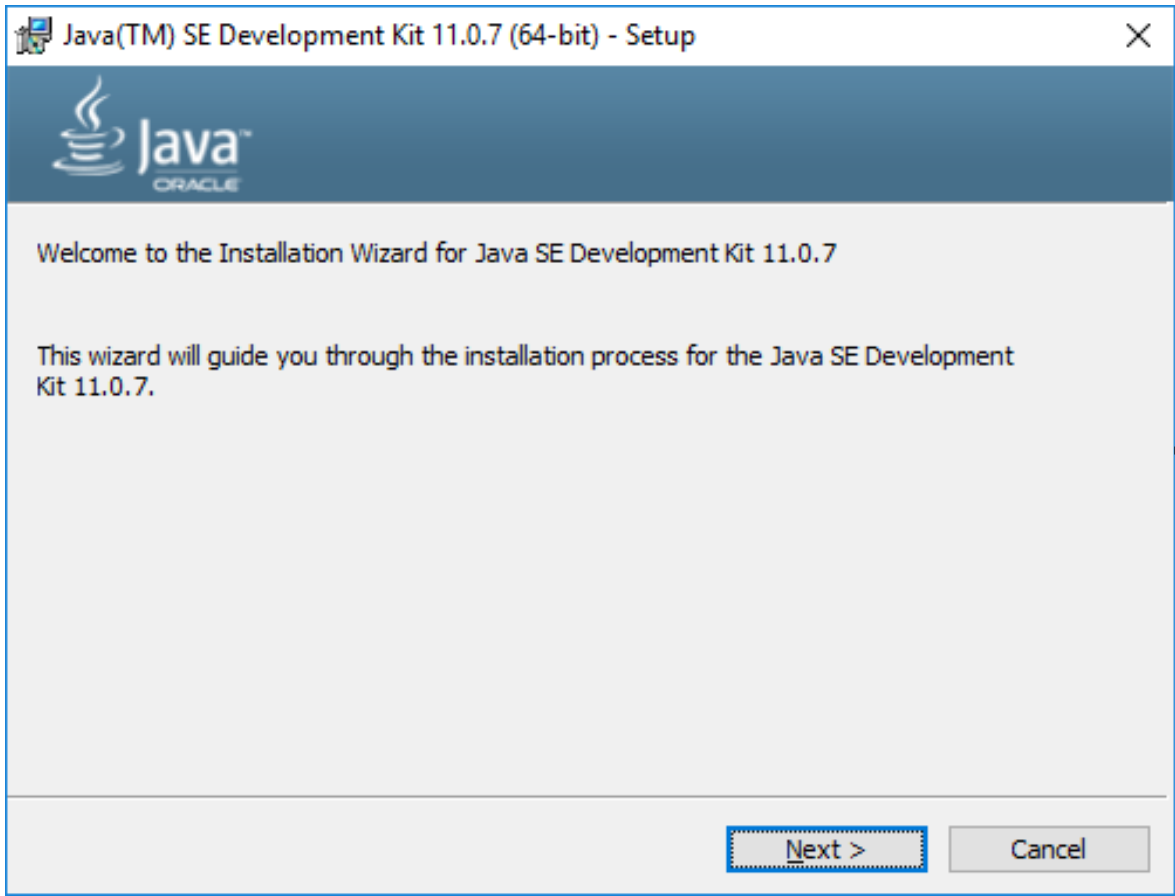
- **Oracle JDK**
- **OpenJDK**

**Per semplicità didattica: OpenJDK LTS (es. 17 o 21)**

**Passaggi:**

1. **Vai sul sito ufficiale**
2. **Scarica la versione LTS**
3. **Scegli il tuo sistema operativo (Windows / macOS / Linux)**
4. **Esegui il file di installazione**





JDK 17 General-Availability Release

This page provides production-ready open-source builds of the Java Development Kit, version 17, an implementation of the Java SE 17 Platform under the GNU General Public License, version 2, with the Classpath Exception.

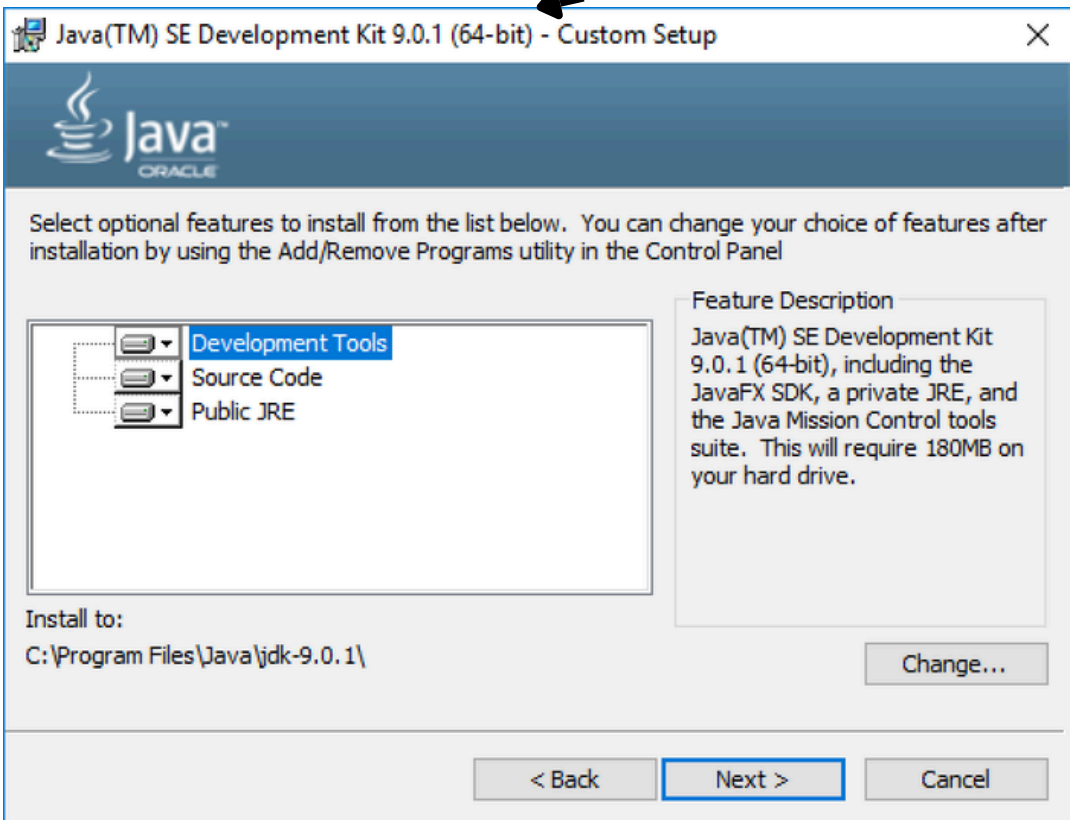
Commercial builds of JDK 17 from Oracle, under a [non-open-source license](#), can be found at the [Oracle Technology Network](#).

Documentation

- [Features](#)
- [Release notes](#)
- [API Javadoc](#)

Builds

Linux/AArch64	tar.gz (sha256)	185360648 bytes
Linux/x64	tar.gz (sha256)	186661523
macOS/AArch64	tar.gz (sha256)	181735146
macOS/x64	tar.gz (sha256)	184007871
Windows/x64	zip (sha256)	185665886



## Verificare l'installazione

**Aprire il terminale (PowerShell ) e digitare:**

**1.java -version**

**e poi:**

**1.javac -version**

**Se tutto è corretto, vedrai la versione installata.**

**⚠ Se javac non viene riconosciuto → problema di PATH.**



## **Aprire VS Code → Extensions (Ctrl+Shift+X)**

### **Installare:**

- **Extension Pack for Java**

### **Include:**

- **Language Support**
- **Debugger**
- **Maven**
- **Test Runner**



## Creare un file Main.java

### Esempio:

```
1. public class Main {  
2.     public static void main(String[] args) {  
3.         System.out.println("Hello Java!");  
4.     }  
5. }
```

**Premere il tasto, angolo alto DX, Run.**



## **Le principali unicità di Java**

### **Portabilità tramite JVM (Write Once, Run Anywhere)**

**Una delle caratteristiche distintive di Java è la portabilità garantita dalla Java Virtual Machine (JVM).**

**Il codice Java non viene compilato direttamente in codice macchina, ma in bytecode, eseguito poi dalla JVM su qualsiasi sistema operativo che la supporti.**

**Questo permette di sviluppare un'applicazione una sola volta ed eseguirla su Windows, macOS o Linux senza modifiche al codice sorgente, rendendo Java particolarmente adatto a contesti enterprise e ambienti eterogenei.**



## **Modello Object-Oriented puro e strutturato**

**Java è progettato fin dall'inizio come linguaggio orientato agli oggetti.**

**Tutto ruota intorno al concetto di classe e oggetto, con un modello chiaro basato su incapsulamento, ereditarietà e polimorfismo.**

**Questa impostazione favorisce modularità, riusabilità e manutenibilità del codice, caratteristiche fondamentali nello sviluppo di software complesso e di lunga durata, come sistemi bancari o applicazioni enterprise.**



## **Gestione automatica della memoria (Garbage Collection)**

**A differenza di linguaggi come C o C++, Java non richiede la gestione manuale della memoria.**

**La JVM integra un Garbage Collector che libera automaticamente gli oggetti non più referenziati.**

**Questo riduce drasticamente errori come memory leak o dangling pointer, aumentando sicurezza e stabilità dell'applicazione.**



**È una caratteristica chiave per ambienti server e applicazioni ad alta disponibilità.**



## **Ecosistema enterprise maturo e standardizzato**

**Java dispone di un ecosistema estremamente consolidato, con librerie standard robuste e framework professionali come Spring.**

**È uno dei linguaggi più utilizzati in ambito enterprise, backend e sistemi distribuiti.**

**L'elevato livello di standardizzazione e retrocompatibilità lo rende una scelta stabile per progetti a lungo termine, soprattutto in contesti aziendali dove affidabilità e scalabilità sono priorità assolute.**

