

tree school

Modulo 1

Introduzione a Java e ambiente di sviluppo

Melvin Massotti



Menù del giorno

Info generali

- Un paio di info generali sul corso

Il linguaggio Java

- La storia, caratteristiche e punti di forza
- Il compilatore Java
- JVM
- JDK (Java Development Kit)
- JRE (Java Runtime Environment)

Ambiente di sviluppo e Primi passi

- Le versioni di Java e la scelta dell'IDE
- Installazione JDK (Windows, Mac, Linux)
- Installazione e Configurazione IDE (Eclipse: Windows, Mac, Linux)
- Creazione di un progetto, compilazione ed esecuzione: Hello World
- Entry Point dell'applicazione

Orario: 9-13 & 14:00-16:00



Info generali

Categorie per gli esercizi

- Ogni modulo del corso è accompagnato da 1 o più **esercizi**
- Ogni esercizio ha una **categoria** che ne annuncia la difficoltà:
- NB: Ogni esercizio è fattibile con gli argomenti spiegati nella teoria

Categorie per gli esercizi



Monopattino



- Gli esercizi «Monopattino» sono quelli base
- Ci sarà sempre almeno un esercizio monopattino
- Questi esercizi saranno introduttivi all'argomento
- È sempre consigliato fare tutti gli esercizi monopattino

Scooter

- Gli esercizi «Scooter» sono quelli avanzati
- In questi esercizi vi verrà chiesto di usare i concetti spiegati a lezione in maniera più approfondita.
- È consigliato fare almeno un esercizio Scooter



Moto

- Gli esercizi «Moto» sono quelli difficili
- Questi esercizi rappresentano una sfida e sono indirizzati verso chi ha completato tutti gli esercizi Scooter
- Questi esercizi non vanno considerati come un plus per chi vuole sviscerare un argomento!





Coding Review

-
- Analizzeremo insieme il codice degli esercizi svolti da 4 – 5 volontari per correggere (ed imparare) insieme
- Col passare delle settimane, **tutti** gli alunni verranno chiamati per la Coding Review
- Nel caso di argomenti più lunghi e complessi, durante il corso potrebbero essere schedate Coding Review aggiuntive per fissare meglio i concetti

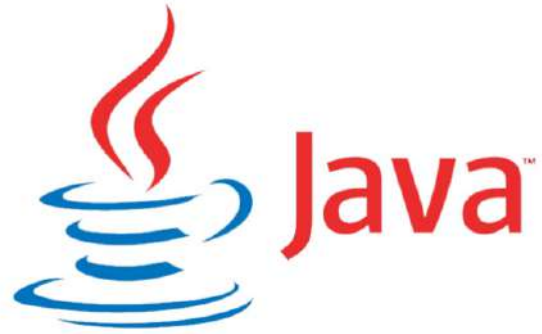
Cos'è la Coding Review

- La Coding Review serve a correggere individualmente difetti nello stile di programmazione e di pensiero per la risoluzione degli esercizi
- Non prendete mai male i nostri consigli, lo facciamo per voi
- La correzione si fa tutti insieme, ma non è una gara



Consigli utili

- Se avete qualsiasi dubbio chiedete e ci ragioniamo insieme
- Collaborate e scambiarevi idee per risolvere gli esercizi
- Durante l'esercitazione, scrivete in chat per chiedermi di venire ad aiutarvi/controllare la vostra soluzione
- L'orario e la durata delle pause li decidiamo insieme in base a quanto state cominciando ad odiare Java ;)



Pillole di informatica

Codice Binario

- Un numero binario è un numero rappresentato in base 2
- Ogni numero è riferito ad un **bit**
- Il bit (dall'inglese "**b**inary **d**igit") è una cifra binaria, ovvero uno dei due simboli del sistema numerico binario, classicamente chiamati zero (0) e uno (1);
- Esempio:

1 0 1 1 0 1

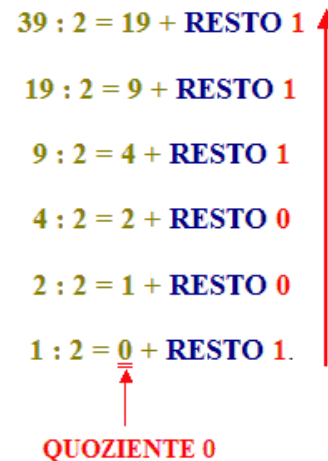
DECIMALE	BINARIO
0	0
1	1
2	10
3	11
4	100
5	101
*****	*****

Da decimale a binario

- Ogni numero decimale può essere pensato in forma binaria come un polinomio a base 2 (esattamente come avviene con il sistema decimale, ma in base 10)
- Esempio: a quanto corrisponde in binario il numero 39?

$$\begin{array}{l} 39 : 2 = 19 + \text{RESTO } 1 \\ 19 : 2 = 9 + \text{RESTO } 1 \\ 9 : 2 = 4 + \text{RESTO } 1 \\ 4 : 2 = 2 + \text{RESTO } 0 \\ 2 : 2 = 1 + \text{RESTO } 0 \\ 1 : 2 = \underline{0} + \text{RESTO } 1. \end{array}$$

↑
QUOZIENTE 0



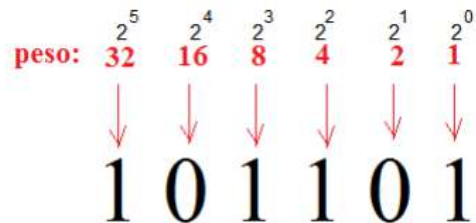
Da decimale a binario

- Ogni numero decimale può essere pensato in forma binaria come un polinomio a base 2 (esattamente come avviene con il sistema decimale, ma in base 10)
- Esempio: a quanto corrisponde in binario il numero 39?
- **Risposta:** 100111

$$\begin{array}{l} 39 : 2 = 19 + \text{RESTO } 1 \\ 19 : 2 = 9 + \text{RESTO } 1 \\ 9 : 2 = 4 + \text{RESTO } 1 \\ 4 : 2 = 2 + \text{RESTO } 0 \\ 2 : 2 = 1 + \text{RESTO } 0 \\ 1 : 2 = \underline{0} + \text{RESTO } 1. \\ \quad \uparrow \\ \text{QUOZIENTE } 0 \end{array}$$

Da binario a decimale

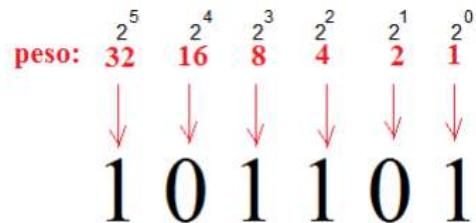
- Ogni numero binario può essere pensato in forma decimale come un polinomio a base 2
- Esempio: a quanto corrisponde in decimale il seguente numero binario?



- NB: iniziare sempre da destra

Da binario a decimale

- Ogni numero binario può essere pensato in forma decimale come un polinomio a base 2
- Esempio: a quanto corrisponde in decimale il seguente numero binario?



- NB: iniziare sempre da destra
- **Risposta:** $2^0 + 2^2 + 2^3 + 2^5 = 45$

Overflow e underflow

- Overflow e underflow sono due fenomeni che avvengono quando proviamo a rappresentare un valore fuori intervallo
- In particolare, otteniamo un **overflow** quando il numero che vogliamo rappresentare è troppo grande, viceversa otteniamo un **underflow** se è troppo piccolo
- Esempio (in codice binario): supponendo di avere a disposizione soltanto 2 bit, proviamo a rappresentare la somma tra **11** (3) e **01** (1), ossia **100**
- Ciò comporta l'overflow di un bit, dunque il risultato ottenuto in realtà sarà **00**, cioè 0!
- Esempio (real life): se assegniamo il valore di 10^{1000} ad una variabile di tipo int o double, otteniamo un overflow perché il limite numerico rappresentabile con un tipo int o double è
- Allo stesso modo, otterremmo un underflow nel caso di 10^{-1000}

«Famosi» overflow

Pac-Man: Level 256 Split Screen Trick

- <https://www.youtube.com/watch?reload=9&v=2K7orNZNGU8>



«Famosi» underflow

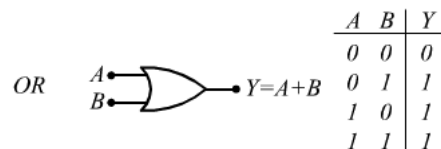
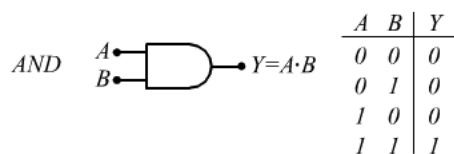
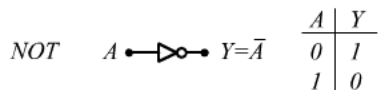
Nuclear Gandhi - Civilization

- <https://www.youtube.com/watch?v=YOg-V4OBZc0>

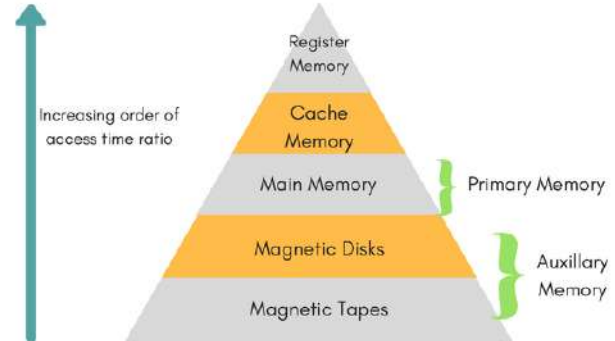
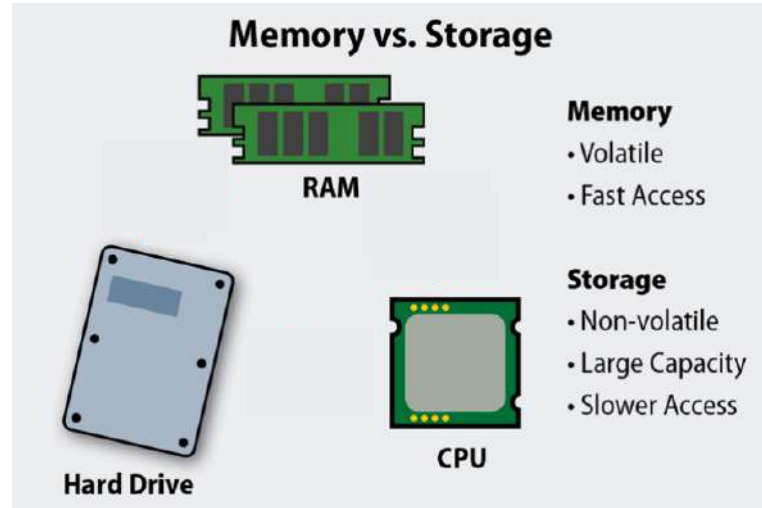


Algebra booleana

- Ramo dell'algebra in cui le variabili possono assumere solamente i valori **vero** e **falso** (valori di verità), generalmente denotati rispettivamente come 1 e 0.
- I più «importanti» operatori dell'algebra booleana sono:



Architettura generale di un elaboratore



Cos'è un linguaggio di programmazione?

- Linguaggio formale che specifica un insieme di istruzioni che possono essere usate per produrre dati in uscita
- Si basa su alcuni concetti fondamentali:
 - **Istruzione:** comando oppure regola descrittiva che, ad ogni esecuzione, lo stato interno del calcolatore (che sia lo stato reale della macchina oppure un ambiente virtuale, teorico, creato dal linguaggio) cambia.
 - **Variabile e costante:** un dato o un insieme di dati, noti o ignoti, già memorizzati o da memorizzare; a una variabile corrisponde sempre, da qualche parte, un certo numero (fisso o variabile) di locazioni di memoria che vengono allocate, cioè riservate, per contenere i dati stessi
 - **Espressione:** una combinazione di variabili e costanti, unite da operatori;

Cos'è un linguaggio di programmazione?

- Linguaggio formale che specifica un insieme di istruzioni che possono essere usate per produrre dati in uscita
- Si basa su alcuni concetti fondamentali:
 - **Strutture dati:** meccanismi che permettono di organizzare e gestire dati complessi.
 - **Strutture di controllo:** che permettono di governare il flusso di esecuzione del programma, alterandolo in base al risultato o valutazione di un'espressione
 - **Funzionalità di input/output:** possibilità di inserire dati da tastiera e visualizzarli in output (stampa a video) attraverso i cosiddetti canali standard (standard input, standard output).
 - **Commenti:** possibilità di inserire dei commenti sul codice scritto, sintatticamente identificati e delimitati, che ne esplichino le funzionalità a beneficio della leggibilità o intelligibilità.

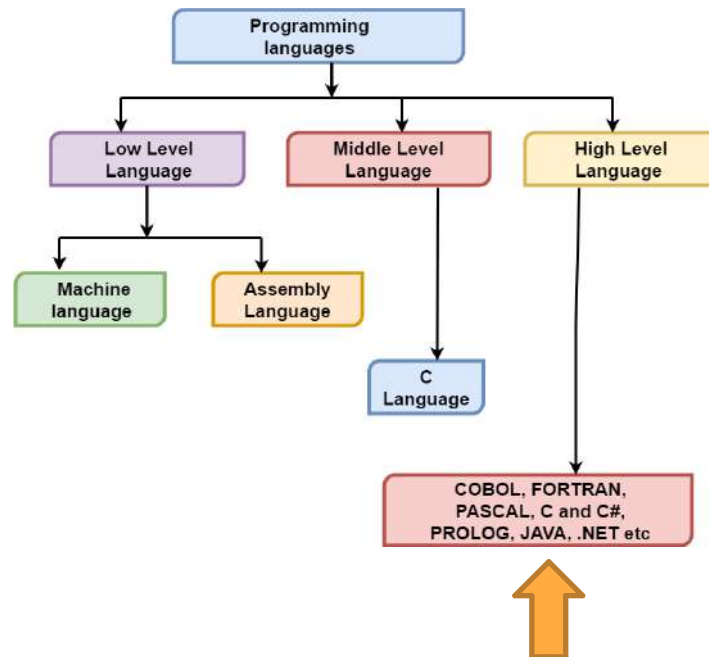
Diversi livelli di astrazione

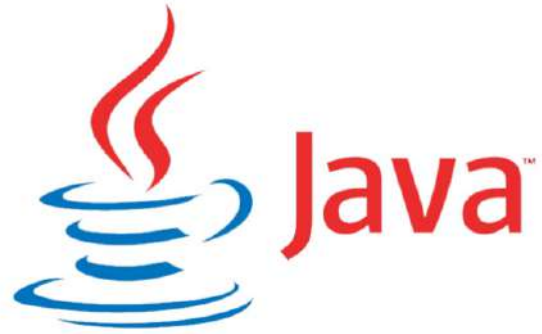
- **Linguaggi ad alto livello**

- Significativa astrazione dai dettagli del funzionamento di un calcolatore e dalle caratteristiche del linguaggio macchina
- Facilità di comprensione per gli esseri umani (discutibile ☺)

- **Linguaggi a basso livello**

- Coincidono con il linguaggio macchina fornendo poca o nessuna astrazione dai dettagli del funzionamento fisico del calcolatore
- Difficoltà di comprensione





Il Linguaggio Java

Storia di java

- Nel 1992 nasce Oak (Quercia) da James Gosling e altri informatici di Sun Microsystems (ora Oracle)
- Per motivi di copyright il nome viene cambiato in Java
- Nato inizialmente per programmare piccoli dispositivi elettronici.
- Java fu annunciato ufficialmente il 23 maggio 1995 a SunWorld.
- L'ultima versione (La Java SE 15) è stata rilasciata il 15 settembre 2020
- Più di 3 miliardi di dispositivi al mondo hanno installato Java.

Caratteristiche di Java

- Un linguaggio di programmazione solido, orientato agli oggetti
- Costruito per essere “sicuro”, cross-platform e internazionale
- Portabile: “WORA” (write once, run anywhere)
- Ha una sintassi “C-style” quindi familiare per molti programmatori
- Mantenuto da una grande azienda, la Oracle
- Di base offre una vasta gamma di **librerie** (mantenute da Oracle) per l'utilizzo di file, strutture dati complesse, rete ecc...

Perché imparare Java



COSTANTEMENTE
AGGIORNATO



AD OGNI
AGGIORNAMENTO È
GARANTITA LA
RETROCOMPATIBILITÀ
(COSA AFFATTO
SCONTATA)



CI SONO UN'INFINITÀ DI
LIBRERIE E FRAMEWORK
DISPONIBILI NEI PIÙ
SVARIATI AMBITI.

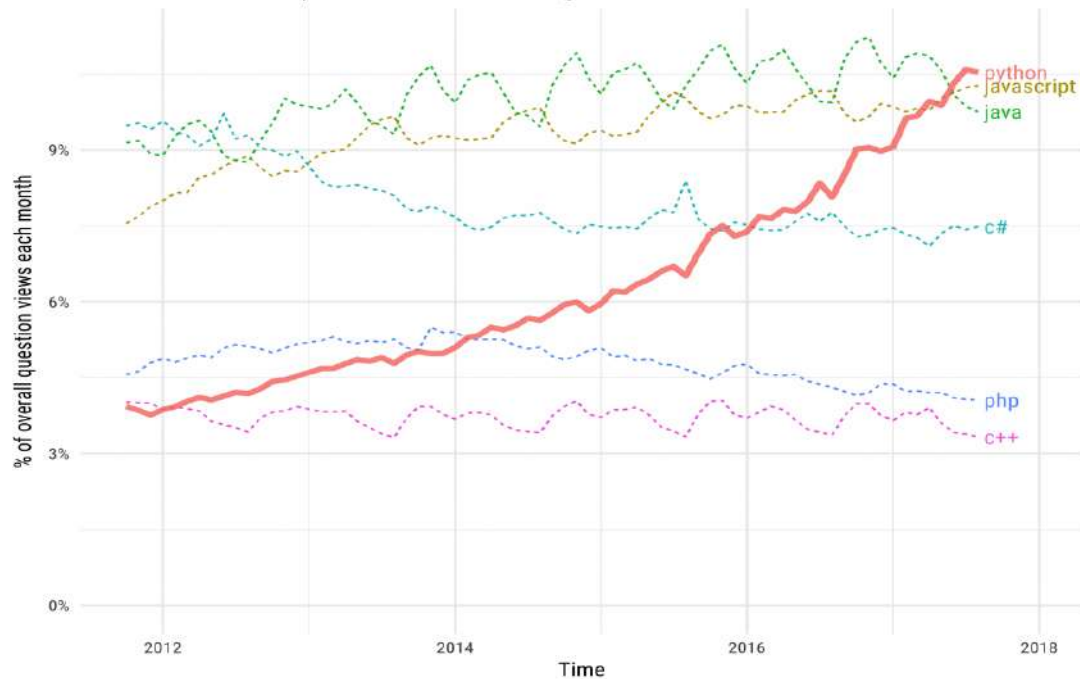


È UNO DEI LINGUAGGI DI
PROGRAMMAZIONE PIÙ
POPOLARI E USATI AL
MONDO

Popolarità dei linguaggi di programmazione

Growth of major programming languages

Based on Stack Overflow question views in World Bank high-income countries



Java e Javascript

- Potreste essere indotti a pensare che questi due linguaggi di programmazione siano molto legati fra loro per via del nome
- In realtà il nome Javascript deriva dalla volontà dei suoi creatori di voler attrarre tramite il nome gli sviluppatori del popolare java
- Sono linguaggi molto diversi tra loro ed anche gli ambienti in cui vengono usati sono molto diversi
- Java e Javascript hanno in comune le stesse cose che hanno in comune un cavallo e un cavalluccio marino



Vi sembrano la stessa cosa?

Dove si utilizza Java

- Grazie alla sua versatilità, Java è stato utilizzato in moltissimi ambiti dell'informatica.
- L'utilizzo principale è per il **back-end** delle piattaforme.
- La maggioranza del software corporate è scritto in Java.
- Il linguaggio di riferimento per sviluppare App **Android** è Java
- Gran parte del software per l'analisi di **big data** è scritta in Java.
- Nonostante java non sia un linguaggio performante, esistono alcuni videogiochi commerciali scritti in Java.

Esempi famosi





Compilazione ed esecuzione

Compilazione ed esecuzione

- La **compilazione** è il processo nel quale il codice scritto in un linguaggio più comprensibile agli esseri umani (ad esempio Java) viene "tradotto" in una serie di **istruzioni macchina**.
- I programmi che fanno questo vengono chiamati **compilatori**
- L'**esecuzione** è il processo tramite il quale un computer esegue le istruzioni macchina seguendone il flusso di esecuzione specificato nel rispettivo codice sorgente

Cosa succede quando eseguo il mio codice Java?

- Un codice Java per poter essere eseguito deve prima essere compilato.
- Il compilatore Java trasforma il codice Java in **bytecode**, un metalinguaggio simile all'assembly.
- Il bytecode viene dato in pasto alla **Java virtual machine (JVM)**, un software in grado di interpretare il bytecode e far eseguire le istruzioni
- Senza una JVM installata non è possibile eseguire del codice Java.
- Questo meccanismo permette ad un file eseguibile Java di essere **indipendente dal sistema operativo** di dove è stato compilato.

JVM (Java Virtual Machine)

- Componente della piattaforma Java che esegue i programmi tradotti in bytecode dopo una prima fase di compilazione
- Definisce i vincoli sintattici e strutturali che i file .class devono rispettare
- Ha un proprio linguaggio macchina: il **bytecode**

Compilazione di un programma Java

- **javac** è il programma che traduce il codice Java in bytecode (file .class), che è un formato intermedio comprensibile dalla JVM.
- La sintassi è la seguente:

javac <NomeFile>.java

- Per maggiori info, è possibile eseguire:

javac -help

Esecuzione di un programma Java

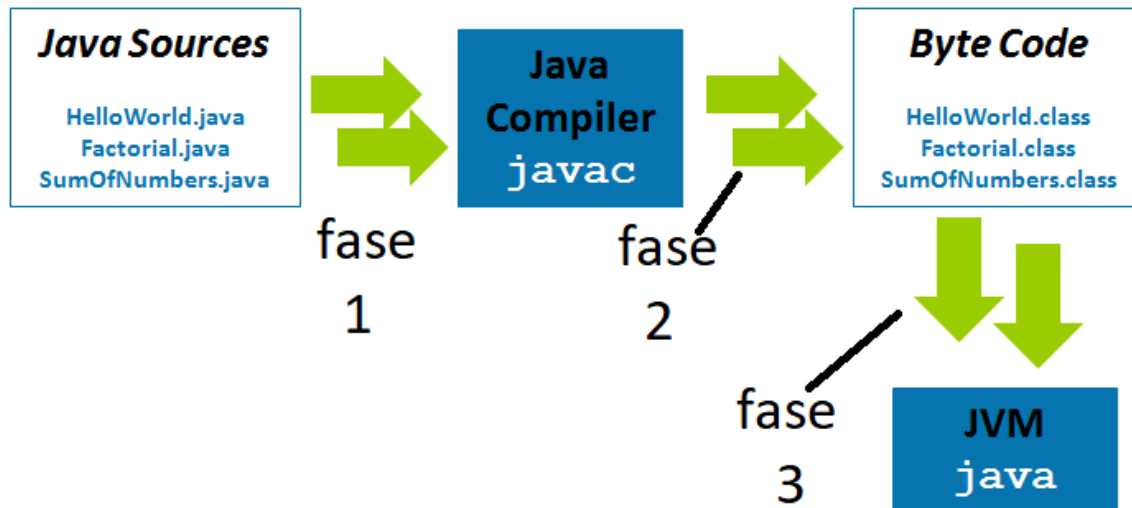
- **java** è il programma che avvia la JVM, la quale si occupa di caricare i file .class (compilati), verificare il bytecode ed eseguirlo.
- La sintassi è la seguente

java <NomeFile>

- Per maggiori info, è possibile eseguire:

java -help

Recap: compilazione ed esecuzione in Java

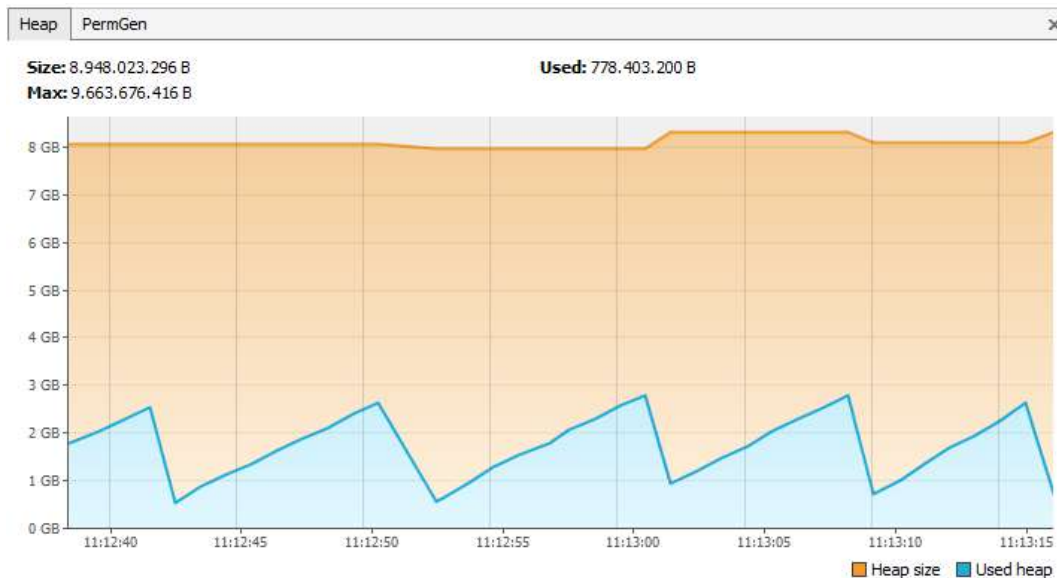


Gestione della memoria

- In Java la gestione della memoria principale (RAM) è automatizzata.
- La JVM di tanto in tanto decide di invocare il garbage collector che ha il compito di "pulire" la memoria principale da tutti gli oggetti rimasti inutilizzati (più avanti questo concetto sarà chiaro).
- Questo sistema è molto comodo per il programmatore poiché non deve preoccuparsi di gestire la memoria come invece avviene in altri linguaggi come il C++.
- Il garbage collector è stato molto criticato in passato perché faceva calare di molto le performance quando veniva invocato, Oracle ha lavorato molto su questo aspetto e ad oggi il garbage collector non è più pesante come un tempo.

Gestione della memoria

Riuscite a vedere
quando entra in
gioco il Garbage
Collector?





Il file Jar

Cos'è un file JAR?

- Formato compresso utilizzato per distribuire applicazioni e librerie Java (**J**ava **A**rchive)
- Deriva dal formato .zip
- I file dati vengono compressi in un unico archivio, rendendone più semplice la distribuzione in una rete.

Creazione di un file JAR

- La sintassi per la creazione di un file Jar è la seguente:

jar cf <nome-file-jar> <nome file o elenco dei file>

- Il parametro 'c' specifica che si sta creando un file JAR
- Il parametro 'f' indica che si vuole specificare il nome del file JAR
- Il parametro <nome-jar-file> è il nome con cui verrà creato l'archivio JAR.
- Il parametro <nome file o elenco dei file> è l'elenco dei file, separati da uno spazio, che verranno inclusi nel file JAR
- Se nell'elenco dei file viene specificato il nome di una directory, il comando jar.exe ne includerà automaticamente tutto il contenuto nel file Jar

Esempio: jar cf miofilejar.jar NomeFile.class

Esecuzione di un file JAR

- La sintassi per l'esecuzione di un file Jar è la seguente:

jar -cp <nome-file-jar> <nome file o elenco dei file>

Esempio: jar cp miofilejar.jar NomeFile

NB: <NomeFile> deve corrispondere al nome della classe principale (quella contenete il main)

Il file Manifest

- Il file Manifest serve a specificare il punto di ingresso dell'applicazione contenuta nel file Jar
- L'aggiunta del file Manifest ci permette di non dover specificare in fase di esecuzione del file Jar il nome della classe principale
- Per convenzione, si utilizza un file Manifest.txt nella cartella principale
- La sintassi del file Manifest è la seguente:

Main-Class: <className>

Esempio: Main-Class: HelloWorld

NB: <className> deve corrispondere al nome della classe principale (quella contenete il main)

Creazione di un file JAR con Manifest

- L'aggiunta del file Manifest ci permette di non dover specificare in fase di esecuzione del file Jar il nome della classe principale
- La sintassi per la creazione di un file Jar è la seguente:

jar cfm <nome-jar-file> <nome file o elenco dei file, tra cui il Manifest>

Esempio: jar cfm miofilejar.jar NomeFile.class

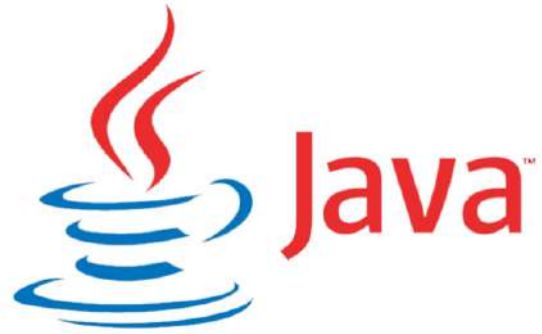
Esecuzione di un file JAR con Manifest

- La sintassi per l'esecuzione di un file Jar contenente un file Manifest è la seguente:

java -jar <nome-file-jar>

Esempio: java -jar miofilejar.jar

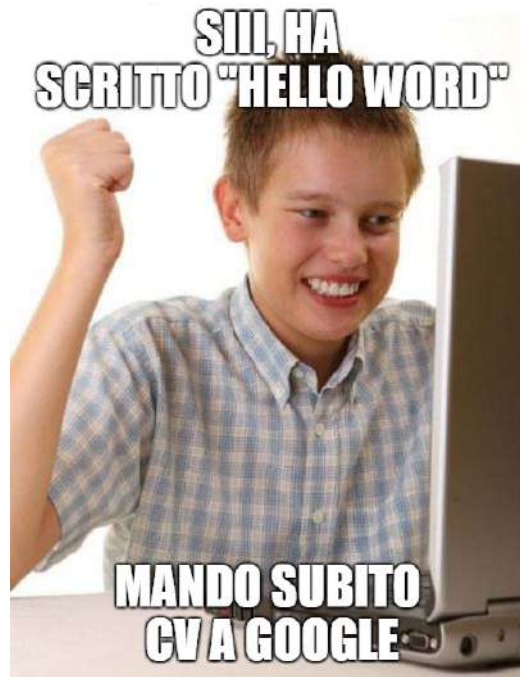
NB: <NomeFile> deve corrispondere al nome della classe principale (quella contenete il main)



Ambiente di sviluppo

JRE e JDK

- **JRE** è l'acronimo di Java Runtime Environment, è un pacchetto software che include tutto ciò che serve per eseguire software java sul proprio computer (inclusa la JVM)
- **JDK** sta invece per Java Development Kit, questo pacchetto non soltanto include tutto ciò che troviamo nel JRE ma ha anche gli strumenti per poter creare nuovo software Java come il compilatore Java.
- Uno dei primi passi che faremo per il laboratorio sarà quello di scaricare e installare il JDK.



Il primo programma in Java

JRE e JDK



Visitate:

<https://www.oracle.com/technetwork/java/javase/downloads/index.html>

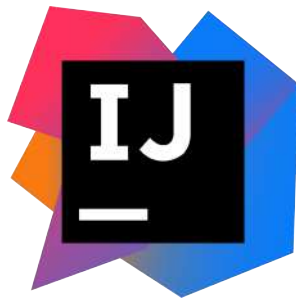
Cercate la versione “Java SE 8u201” e cliccate su JDK.

Accettate la licenza, e scaricate la versione del software per il vostro sistema operativo

Installate Java e siete quasi pronti per scrivere il vostro primo programma!

- Nonostante si possa tranquillamente lavorare con Java utilizzando soltanto un editor di testo e compilando il codice tramite riga di comando, usare un **IDE** (*integrated design environment*) è sicuramente la scelta più comoda e saggia.
- Un IDE è quindi un software che offre aiuti e funzionalità al programmatore nella scrittura del software
- Per esempio in un IDE possiamo trovare come funzionalità il controllo runtime di errori di sintassi nel codice e supporto per l'auto-completamento del codice

Quale IDE scegliere?



- Sono i due IDE per Java più popolari del momento.
- Entrambi sono costantemente aggiornati e mantenuti.
- IntelliJ ha sicuramente più funzionalità rispetto ad Eclipse al costo però di una maggiore pesantezza.
- Per utilizzare IntelliJ in versione base non bisogna comprare una licenza, anche Eclipse è gratuito.
- Potete scegliere liberamente, il consiglio è di utilizzare IntelliJ.



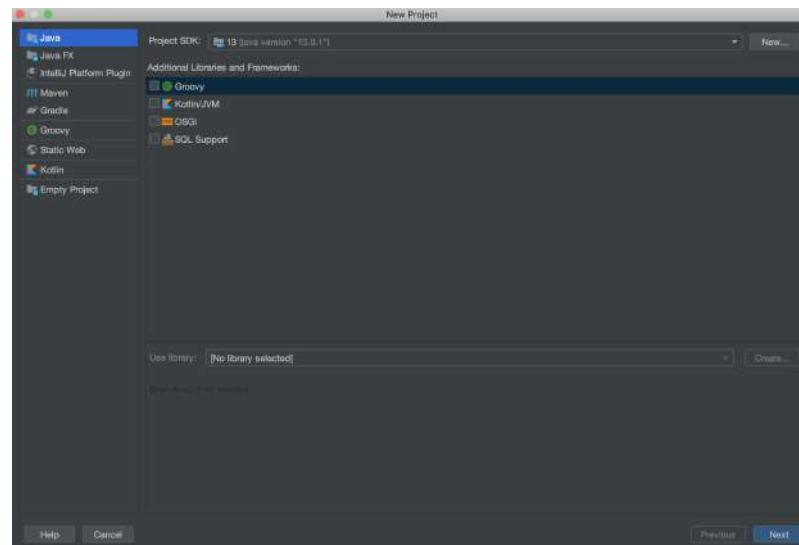
Installazione IntelliJ



- Andate su:
<https://www.jetbrains.com/idea>
- Dovrebbe proporvi in automatico l'ultima versione per il vostro sistema operativo
- Scaricate e installate la versione free (community)

Nuovo progetto in IntelliJ

Cliccate su Create New Project



Il progetto Hello world

- Date come nome al progetto "Hello World" e cliccate «finish».
- Se non ci sono stati problemi dovreste trovare il vostro progetto "Hello World" nella barra laterale a sinistra.
- Espandendola troverete due cartelle, JRE e src.
- In JRE ci sono le librerie base di Java caricate (non avete bisogno di toccare nulla)
- **src** è la cartella dove inserirete il vostro codice.

Ultima cosa



Ogni programma Java per poter essere eseguito deve possedere almeno una classe contenente il metodo **main**

Questo metodo sarà l'accesso per la JVM al nostro programma.

Il metodo main



```
public static void main(String[] args) {  
}
```

C'è una sintassi ben precisa per indicare quale è il metodo **main**. Per il momento limitatevi ad impararla, più avanti nel corso capirete da soli il significato e il perché di queste parole.

Nuova Classe in IntelliJ



Tasto destro => new => Java Class



Hello World

Il nostro obiettivo sarà quello di scrivere un programma che stamperà sulla console (la barra in basso) la scritta "Hello World!«

Chiamate la classe "HelloWorld"

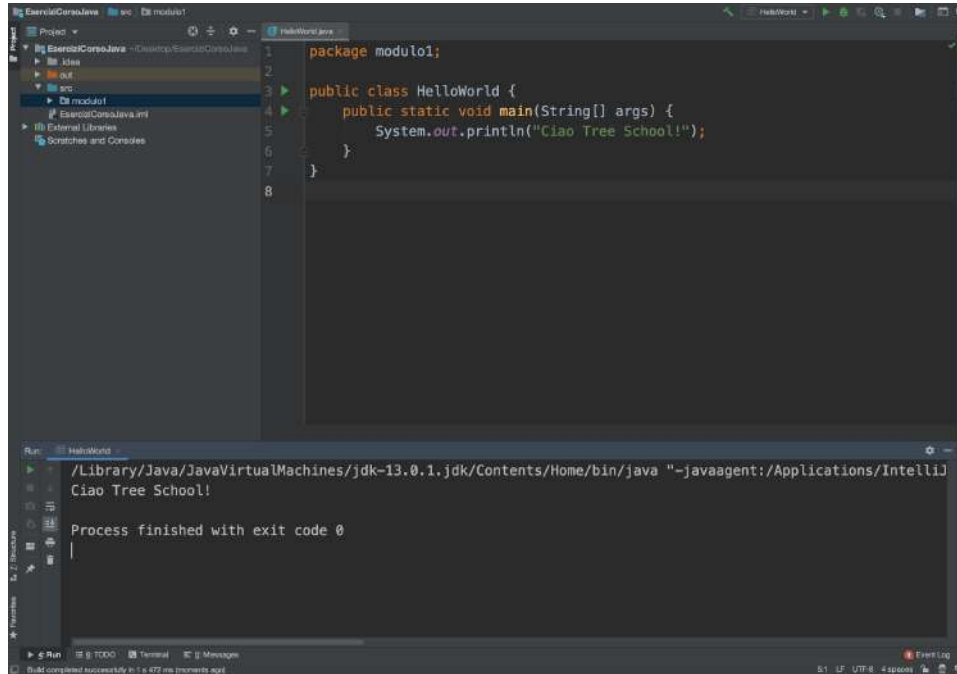
Aggiungete un metodo main copiando il pezzo di codice della slide precedente

L'istruzione che permette di scrivere sulla console è: *System.Out.println();*

Scrivete l'istruzione per scrivere e inserite la stringa "Hello World!" dentro le parentesi tonde.

Ora premete il tasto verde con il play bianco per avviare il vostro programma.

Hello World



The screenshot shows an IDE window with a project named 'EsercizioCorsoJava'. The main editor displays the following Java code:

```
1 package modul01;
2
3 public class HelloWorld {
4     public static void main(String[] args) {
5         System.out.println("Ciao Tree School!");
6     }
7 }
8
```

Below the editor, the 'Run' tab shows the execution output:

```
/Library/Java/JavaVirtualMachines/jdk-13.0.1.jdk/Contents/Home/bin/java "-javaagent:/Applications/IntelliJ
Ciao Tree School!
Process finished with exit code 0
```

The status bar at the bottom indicates 'Build completed successfully in 1 s 472 ms (IntelliJ IDEA)'.

Se il vostro programma ha stampato hello world come in questo esempio avete completato l'esercizio è scritto il vostro primo programma in Java!

Grazie per l'attenzione!

