

2 - Introduzione al linguaggio Java

Programmazione e analisi di dati
Modulo A: Programmazione in Java

Paolo Milazzo

Dipartimento di Informatica, Università di Pisa
<http://www.di.unipi.it/~milazzo>
milazzo@di.unipi.it

Corso di Laurea Magistrale in Informatica Umanistica
A.A. 2013/2014

Sommario

1 Introduzione al linguaggio Java

2 Scrivere, compilare ed eseguire un programma Java

- Il primo programma Java
- Editare, compilare ed eseguire

La Genesi di Java

Java è un linguaggio di programmazione nato all'inizio degli anni novanta da un gruppo di lavoro della **Sun Microsystems** guidato da **James Gosling**

Inizialmente concepito per scrivere programmi per il controllo di elettrodomestici (TV, frigorifero,...)

- linguaggio (relativamente) **semplice da usare**
- capace di essere eseguito su **diversi tipi di processori**
- che **non richiedesse compilatori o interpreti troppo sofisticati** (i produttori degli elettrodomestici non avrebbero investito risorse in quel settore)

L'idea fu di introdurre un unico, semplice **linguaggio intermedio** (chiamato **byte-code**) per il quale potessero facilmente essere scritti interpreti ad-hoc

Il byte-code Java (1)

Il linguaggio Java si basa quindi su un approccio che combina compilazione (in byte-code) e interpretazione (del byte-code)

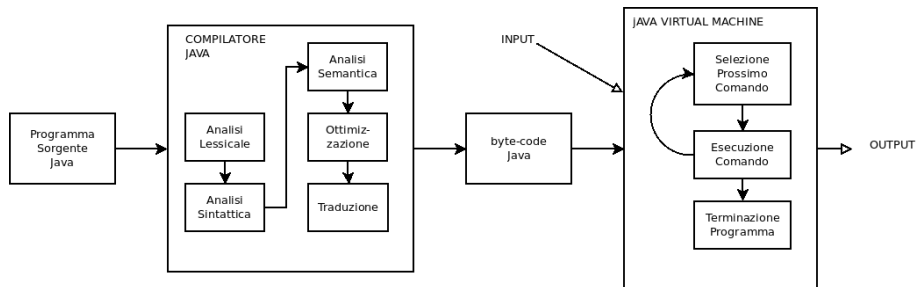
Il byte-code può essere visto come l'assembly di una **macchina virtuale**, un calcolatore ipotetico che ha caratteristiche simili (semplificate) a quelle delle architetture hardware più comuni

- è un linguaggio di basso livello (come l'assembly)
- non è legato ad una particolare architettura hardware

L'interprete del byte-code Java è detto **Java Virtual Machine (JVM)**

Il byte-code Java (2)

L'approccio compilazione+interpretazione schematicamente:



Java e Internet

Dopo breve tempo ci si rese conto che Java poteva essere usato per distribuire applicazioni su **Internet**

- il byte-code poteva essere distribuito via Web ed essere eseguito sui computer degli utenti
- essenziale l'indipendenza dalla piattaforma hardware
- successo delle **applet Java**: programmi Java eseguibili dentro al browser Web (la JVM installata come plug-in del browser)

Con il tempo altre tecnologie soppiantano Java nell'ambito di Internet (e.g. JavaScript)

Java rimane comunque uno tra i principali linguaggi per lo sviluppo di **applicazioni desktop e distribuite**, in particolare in ambiente aziendale (**enterprise**)

Java e gli altri linguaggi

(per chi conosce C e C++)

Il linguaggio Java ha una sintassi che si ispira ai linguaggi C e C++

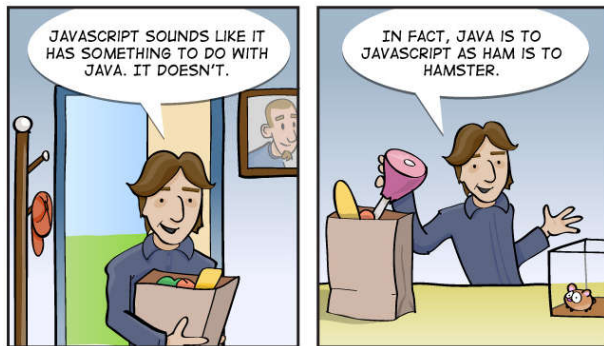
- erano probabilmente i linguaggi più usati all'inizio degli anni 90

Java, come il C++, è un **linguaggio a oggetti** (o **object-oriented**)

- Un programma può essere strutturato come un insieme di oggetti che interagiscono l'uno con l'altro (vedremo...)
- La gestione degli oggetti (e non solo) è semplificata rispetto a C++

Java vs JavaScript (1)

(per chi conosce JavaScript)



Java vs JavaScript (2)

JavaScript è un linguaggio che serve per scrivere **applicazioni web client-side** da eseguire all'interno del browser

Rispetto a JavaScript, il linguaggio Java:

- Prevede una fase di compilazione che effettua numerosi **controlli**
- Prevede **regole sintattiche** più forti (e.d. il ; alla fine di ogni comando)
- E' un linguaggio **fortemente tipato**:
 - ▶ il programmatore è tenuto a specificare **il tipo** di ogni variabile, e il compilatore richiede e garantisce che i valori di tali variabili verranno sempre usati in modo coerente rispetto al tipo
- Fa un utilizzo degli oggetti molto diverso (basato su classi)

Insomma... Java è molto più rigoroso di JavaScript... quindi meglio si presta a scrivere **applicazioni complesse e strutturate**

Sommario

1 Introduzione al linguaggio Java

2 Scrivere, compilare ed eseguire un programma Java

- Il primo programma Java
- Editare, compilare ed eseguire

Il primo programma Java (1)

Il seguente programma visualizza un semplice saluto (Hello World!)

```
public class HelloWorld {  
    public static void main(String[] args) {  
        //visualizza un messaggio di saluto  
        System.out.println("Hello World!");  
    }  
}
```

Il primo programma Java (2)

```
public class HelloWorld { .... }
```

... dice che stiamo definendo la **classe** HelloWorld

- Un programma Java è costituito da un insieme di classi (almeno una)
- `public` significa che questa classe è pubblica: può essere utilizzata da qualunque altra classe del programma
- Il contenuto della classe è racchiuso tra parentesi graffe

Il primo programma Java (3)

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

.... definisce un **metodo** della class HelloWorld

- Un metodo è una funzionalità della classe messa disposizione del resto del programma o di altre parti (altri metodi) della stessa classe
- Questo metodo è chiamato `main` (principale) ed è un metodo speciale. Viene subito **eseguito all'inizio** del programma.
- `public` significa che questo metodo può essere usato da altre classi
- `static`, `void` e `String[] args` li capiremo più avanti....
- Il **corpo** del metodo è racchiuso tra le parentesi graffe

Il primo programma Java (4)

```
//visualizza un messaggio di saluto
```

.... è un **commento**

- Viene trascurato dal compilatore Java
- Serve solo per rendere più comprensibile il programma
- E' un commento tutto ciò che si trova a destra di // (una sola riga)
- E' un commento anche tutto ciò che si trova tra /* e */ (anche su più righe). Ad esempio:

```
/* Questo e' un esempio di commento su  
due righe */
```

Il primo programma Java (5)

```
System.out.println("Hello World!");
```

.... è un **comando** che visualizza il messaggio Hello World

- `System.out` è un **oggetto** che rappresenta il canale di output standard del sistema (la console...)
- Un **oggetto** è un'entità attiva che corrisponde a una determinata classe (vedremo...)
- `println` è un metodo dell'oggetto `System.out` che stampa un messaggio e va a capo
- **come tutti i comandi**, `println` deve essere terminato con **punto e virgola ;**
- "Hello World!" è una **stringa**, ossia una sequenza di caratteri alfanumerici
- La stringa "Hello World!" viene passata come **parametro** (tra parentesi) al metodo `println`

Struttura di base

Per un po' di tempo i programmi Java che considereremo avranno sempre la struttura

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

ossia:

- Una sola classe (con nome arbitrario)
- Il solo metodo `main` (scritto esattamente come nell'esempio)
- Il corpo del `main` conterrà tutti i comandi del programma

Sommario

- 1 Introduzione al linguaggio Java
- 2 Scrivere, compilare ed eseguire un programma Java
 - Il primo programma Java
 - Editare, compilare ed eseguire

Editare un programma Java

Per scrivere un programma Java si può usare un qualunque **editor** di testi

Tra i più semplici:

- su Linux: gedit
- su Windows: il “blocco note”
(o l'ottimo “Notepad++” – <http://notepad-plus-plus.org/>)

E' sufficiente aprire l'editor, digitare il programma e salvarlo (in una opportuna directory) con il nome

```
<nomeclasse>.java
```

Quindi la classe HelloWorld vista prima dovrà essere salvata come HelloWorld.java

Compilare ed eseguire un programma Java (1)

Per compilare ed eseguire un programma Java avremo bisogno di un **compilatore Java** e di una **Java Virtual Machine (JVM)**

Entrambi questi strumenti sono forniti dal **Java Development Kit (JDK)**.

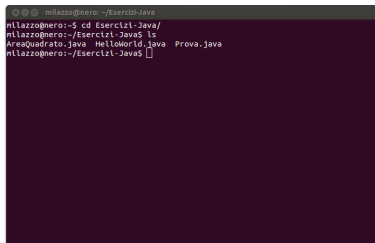
Sul sito web del corso trovate i link alle pagine che vi consentono di scaricare il JDK

Un modo per eseguire il compilatore java e la JVM è tramite la **console** di sistema

- Su Linux (e su MacOS) si chiama **Terminale**
- Su Windows si chiama **Prompt dei comandi**

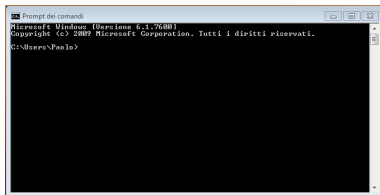
Compilare ed eseguire un programma Java (2)

Primo passo: aprire il terminale/prompt dei comandi



```
milazzo@nero: ~/Esercizi-Java
milazzo@nero:~$ cd Esercizi-Java/
milazzo@nero:~/Esercizi-Java$ ls
AreaQuadrato.java HelloWorld.java Prova.java
milazzo@nero:~/Esercizi-Java$
```

Terminale



```
Prompt dei comandi
Microsoft Windows [Versione 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. Tutti i diritti riservati.
C:\Users\Paolo>
```

Tutti i programmi – Accessori – Prompt dei comandi

Secondo passo: spostarsi nella cartella (directory) che contiene il programma usando i comandi `ls` (Linux) o `dir` (Windows) e `cd` (Linux/Win)

- `ls` e `dir` mostrano il contenuto della directory corrente
- `cd nome_dir` consente di entrare dentro la directory con nome `nome_dir`
- `cd ..` consente di uscire dalla directory corrente

Compilare ed eseguire un programma Java (3)

Terzo passo: Raggiunta la directory che contiene il programma, si può eseguire il **compilatore Java** tramite il comando **javac**

```
javac <nomeclasse>.java
```

Quindi, nel caso del programma **HelloWorld** dovremo digitare `javac HelloWorld.java` (N.B. il file deve esistere nella directory!)

- Nota: Se `javac` non funziona può darsi che sia necessario impostare la variabile di sistema `PATH` (vedere note installazione di Java sul sito del corso)

Eventuali errori nel programma vengono segnalati ora!

Se il programma non contiene errori, il risultato della compilazione è il file `<nomeclasse>.class` (nell'esempio: `HelloWorld.class`) che contiene il byte-code

Compilare ed eseguire un programma Java (4)

Quarto passo: Bisogna ora eseguire la **Java Virtual Machine** tramite il comando **java**

```
java <nomeclasse>
```

Quindi, nel caso del programma **HelloWorld** dovremo digitare `java HelloWorld` e il programma ci risponderà "Hello World!"

Compilare ed eseguire un programma Java (5)

Riassumendo...

The screenshot displays a Windows desktop environment with three main windows:

- Code Editor:** A Notepad window titled "HelloWorld.java - Blocco note" containing the following Java code:

```
public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello World!");  
    }  
}
```
- Command Prompt:** A "PROMPT dei comandi" window showing the execution of the following commands:

```
Microsoft Windows [Versione 6.1.7600]  
Copyright (c) 2009 Microsoft Corporation. Tutti i diritti riservati.  
C:\Users\Paolo>cd Documents  
C:\Users\Paolo\Documents>cd HelloWorld  
C:\Users\Paolo\Documents\HelloWorld>dir  
Il volume nell'unità C non ha etichetta.  
Numero di serie del volume: 6849-7E0D  
  
Directory di C:\Users\Paolo\Documents\HelloWorld  
30/09/2013 11:20 <DIR>  
30/09/2013 11:20 <DIR>  
30/09/2013 11:07             126 HelloWorld.java  
                           126 byte  
2 Directory 10.903.085.056 byte disponibili  
C:\Users\Paolo\Documents\HelloWorld>javac HelloWorld.java  
C:\Users\Paolo\Documents\HelloWorld>java HelloWorld  
Hello World!  
C:\Users\Paolo\Documents\HelloWorld>
```
- File Explorer:** A "Raccolta Documenti" window showing the contents of the "HelloWorld" folder. It contains a table with the following data:

Nome	Ultima modifica	Tipo	Dimensione
HelloWorld.class	30/09/2013 11:23	File CLASS	1 KB
HelloWorld.java	30/09/2013 11:07	File JAVA	1 KB

The taskbar at the bottom shows the Start button, Internet Explorer, File Explorer, and several application icons. The system tray on the right indicates the time as 11:24 on 30/09/2013.

Uso della shell

Quando abbiamo usato la console di sistema (Terminale o Prompt dei comandi) abbiamo in realtà interagito con un programma detto **shell**

- La **shell** è il programma che esegue interattivamente comandi di sistema (`cd`, `dir`, `ls`, ...)

Tramite la shell si possono fare un sacco di cose (creare/rimuovere directory, eseguire programmi, controllare i programmi attivi, ecc...)

- Fino a un po' di anni fa la shell era l'unico mezzo per usare un computer

Esistono diverse shell (che eseguono comandi diversi)

- Su Linux la più comune si chiama **bash**
- Su Windows l'unica in pratica disponibile deriva dall'**MS-DOS**

Trovate riferimenti a **guide** su bash e MS-DOS **nel sito web del corso!**