

Java: Primo Approccio

Walter Cazzola

Dipartimento di Informatica e Comunicazione
Università degli Studi di Milano

Outline

- 1 **Il Processo di Sviluppo del Software.**
 - Scrivere un Programma Java
 - Compilazione di un Programma Java
 - Esecuzione di un Programma Java
- 2 **Librerie e Packages.**
 - Packages: Definizione e Uso
 - Java (SDK) Standard Library: Date
 - Packages Non Standard: prog.io.*
- 3 **Esempi ed Esercizi.**
 - Calcolo dell'Area di un Rettangolo
 - Conversione dei Gradi Celsius ($^{\circ}\text{C}$) in Fahrenheit ($^{\circ}\text{F}$)
 - Esercizi

Scopo della Lezione.

Realizzare dei semplici programmi scritti in Java.

Esercitarsi nelle operazioni necessarie per passare dalla scrittura di codice Java all'esecuzione del programma correlato.

Utilizzare le classi di I/O.

Familiarizzare con il concetto di variabile.

Il Processo di Sviluppo del Software.

Passo 1: Scrivere il programma.

- SW necessario: un qualsiasi text editor, es. vim.

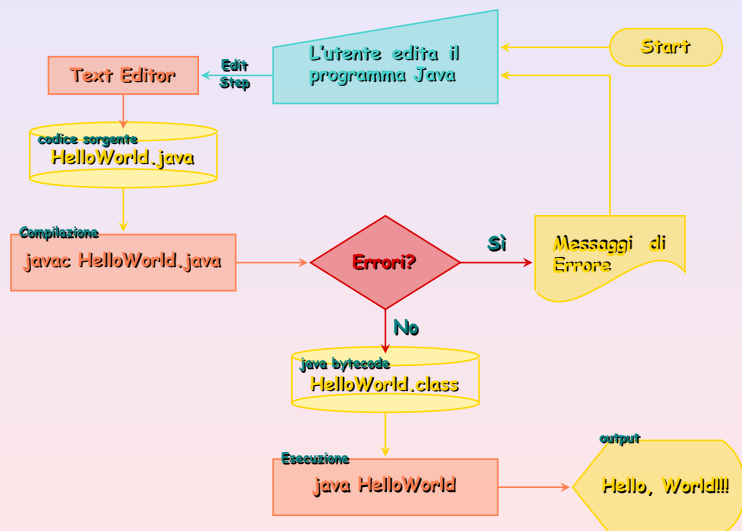
Passo 2: Compilare il programma

- SW necessario: Java Development Kit (JDK)
- Comando: `javac <nome programma>.java`

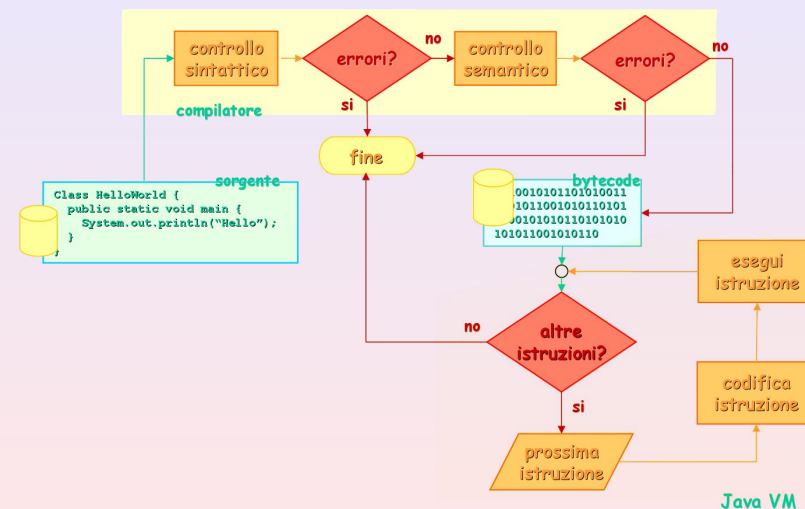
Passo 3: Eseguire il programma

- SW necessario: JDK o Java Runtime Environment (JRE)
- Comando: `java <nome classe principale>`

Compilazione ed Esecuzione.



Macchina Virtuale.



Scrivere un Programma Java.

Il codice sorgente del programma deve essere scritto con un text editor e comunque salvato come un file di testo (ASCII).

Il file contenente il codice sorgente deve chiamarsi come la classe che implementa e con estensione .java

Nota. I nomi dei file e delle classi sono case sensitive (cioè Pippo e piPpo sono nomi diversi).

Compilare i Programmi Java.

La fase di compilazione traduce il sorgente del programma Java in **bytecode**.

- il bytecode è indipendente dalla piattaforma.

Comando JDK: javac HelloWorld.java.

Quando la compilazione termina con successo vengono creati i file contenenti, ognuno, il bytecode di una delle classi definite nel codice compilato, ad es. HelloWorld.class

Eeguire un Programma Java.

Il file contenente il bytecode della classe deve essere caricato in memoria ed interpretato dalla **Java Virtual Machine (JVM)**.

Comando JDK: `java HelloWorld`.

Nota. Per poter essere eseguita, una classe deve definire un metodo di nome `main()`.

Convenzioni sulla Scrittura dei Programmi.

I programmi devono seguire le seguenti regole:

- tutti i file relativi ad un programma risiederanno in una sottodirectory;
- il nome di un programma sarà scritto in caratteri minuscoli, ad eccezione delle iniziali delle parole.

Inoltre, gli elementi del programma **DEVONO** avere nomi significativi.

- Programma `e` `UnProgrammaBellissimo` sono nomi leciti; mentre
- `PROGRAMMA` `e` `Unprogrammabellissimo` **no**.

HelloWorld.java

Consideriamo:

```
/* Questo è il nostro primo programma scritto in Java */  
  
class HelloWorld {  
    public static void main(String args[]) {  
        System.out.println("Hello, World!!!");  
    }  
} // Analizziamolo!!!
```

Compiliamolo:

```
[14:22]cazzola@ulik:~> cd HelloWorld  
[14:22]cazzola@ulik:~/HelloWorld> ls  
HelloWorld.java  
[14:23]cazzola@ulik:~/HelloWorld> javac HelloWorld.java  
[14:23]cazzola@ulik:~/HelloWorld> ls  
HelloWorld.class HelloWorld.java
```

Eseguiamolo:

```
[14:24]cazzola@ulik:~/HelloWorld> java HelloWorld  
Hello, World!!!  
[14:25]cazzola@ulik:~/HelloWorld>
```

HelloWorld.java

```
/* Questo è il nostro primo programma scritto in Java */  
  
class HelloWorld {  
    public static void main(String args[]) {  
        System.out.println("Hello, World!!!");  
    }  
} // Analizziamolo!!!
```

Commenti.

- Il testo racchiuso tra `/*` e `*/` rappresenta un commento e può estendersi su più righe.
- Ciò che segue `//` rappresenta un commento e viene ignorato fino alla fine della riga.

Nota. La presenza di commenti non modifica il comportamento del programma . . . ma ne aumenta la leggibilità.

HelloWorld.java

```
/* Questo è il nostro primo programma scritto in Java */
```

```
class HelloWorld {  
    public static void main(String args[]) {  
        System.out.println("Hello, World!!!");  
    }  
} // Analizziamolo!!!
```

Commenti a parte, il programma è delimitato:

- dalla parola chiave **class**;
- seguita dal nome del programma e
- da una coppia di parentesi graffe.

In generale, le parentesi graffe delimitano blocchi di codice, indentati conseguentemente.

HelloWorld.java

```
/* Questo è il nostro primo programma scritto in Java */
```

```
class HelloWorld {  
    public static void main(String args[]) {  
        System.out.println("Hello, World!!!");  
    }  
} // Analizziamolo!!!
```

Il blocco **main()**:

- rappresenta il blocco di istruzioni che viene eseguito quando si esegue il programma;
- per ora non ci interessiamo delle parole chiave che precedono e seguono **main()**.

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

- quando eseguita stampa sullo schermo il messaggio specificato tra virgolette andando a capo;
- **System.out.print()** ha lo stesso comportamento ma non va a capo.

HelloWorldPrint.java

```
/* Questo è il nostro secondo programma scritto in Java */
```

```
class HelloWorldPrint {  
    public static void main(String args[]) {  
        System.out.print("Hello, World!!!");  
    }  
}
```

Compiliamolo ed Eseguiamolo:

```
[18:25]cazzola@ulik:~> cd HelloWorldPrint  
[18:26]cazzola@ulik:~/HelloWorldPrint> javac HelloWorldPrint.java  
[18:26]cazzola@ulik:~/HelloWorldPrint> java HelloWorld  
Hello, World!!![18:27]cazzola@ulik:~/HelloWorldPrint>
```

Come vedete il prompt è apparso sulla stessa riga dell'output del programma.

Librerie

Java mette a disposizione un insieme di comandi e di oggetti deputati a risolvere particolari compiti.

Oggetti con funzionalità simili o collegate sono raggruppati in un insieme che viene chiamato **package**. Alcuni package possono far parte di altri package.

Il pacchetto di sviluppo JDK mette a disposizione diversi package per adempiere i compiti più svariati.

Package

Per poter usare un package è necessario importarlo nel programma, tramite l'istruzione **import**

- **import** java.util.*
permette di usare tutti i servizi offerti dal package java.util.
- **import** java.util.Date
permette di usare esclusivamente i servizi offerti dalla classe Date.

Alcuni package, come java.lang, vengono importati implicitamente.

HelloDate.java

```
/* HelloDate stampa la data corrente */  
  
import java.util.Date;  
  
public class HelloDate {  
    public static void main(String args[]) {  
        System.out.print("La data Corrente è: ");  
        System.out.println(new Date());  
    }  
}
```

Compiliamolo ed Eseguiamolo:

```
[22:09]cazzola@ulik:~/HelloDate>javac HelloDate.java  
[22:10]cazzola@ulik:~/HelloDate>java HelloDate  
La data Corrente è: Mon Nov 06 22:10:48 CET 2006  
[22:10]cazzola@ulik:~/HelloDate>
```

L'output conterrà la data (giorno, mese e anno in formato Americano) e l'orario corrente (ora e fuso orario).

HelloDate.java: Come Funziona?

import java.util.Date
Mette a disposizione i servizi offerti dalla classe Date.

new Date()
Istanza un nuovo oggetto della classe Date e ne ritorna un riferimento.

System.out.println()
È programmato per riconoscere i riferimenti alle istanze di Date che gli vengono passate come argomento e le tratta in modo opportuno (cioè stampa la data corrente).

Un Package Importante: prog.io.*

Per gestire l'input da tastiera e l'output a video si useranno rispettivamente le classi:

- ConsoleInputManager, e
- ConsoleOutputManager

che fanno parte del package prog.io distribuito con il libro di testo.

Attenzione!!!

Il package prog.io non fa parte della distribuzione standard di Java. Per usarlo occorre:

- copiare il package dal CD del libro sul proprio computer (opzionale se si usa la distribuzione usata in laboratorio);
- modificare il contenuto della variabile di ambiente CLASSPATH (su linux export CLASSPATH=\$CLASSPATH:«path libreria»).

HelloWorldModificato.java

```
import prog.io.ConsoleOutputManager;

class HelloWorldModificato {
    public static void main(String args[]) {
        ConsoleOutputManager video = new ConsoleOutputManager();
        video.println("Hello, World!!!");
    }
}
```

Compiliamolo ed Eseguiamolo:

```
[22:12]cazzola@ulik:~/HelloWorldModificato>javac HelloWorldModificato.java
[22:12]cazzola@ulik:~/HelloWorldModificato>java HelloWorldModificato
Hello, World!!!
[22:12]cazzola@ulik:~/HelloWorldModificato>
```

L'output è lo stesso, ma è stato prodotto in modo differente.

AreaRettangolo.java

```
public class AreaRettangolo {
    public static void main(String args[]) {
        ConsoleOutputManager video = new ConsoleOutputManager();
        int base=3;
        int altezza=4;
        video.print("L'area è: ");
        video.println(base*altezza);
    }
}
```

Compiliamolo ed Eseguiamolo:

```
[22:14]cazzola@ulik:~/AreaRettangolo>javac AreaRettangolo.java
[22:14]cazzola@ulik:~/AreaRettangolo>java AreaRettangolo
L'area è: 12
[22:14]cazzola@ulik:~/AreaRettangolo>
```

AreaRettangolo.java: Alcuni Dettagli

`println()`

L'argomento è una moltiplicazione, questa verrà valutata prima del metodo stesso ed il risultato sarà l'argomento del metodo `println()`.

`base` e `altezza`

Il programma calcola l'area usando dei dati numerici che memorizza in due aree di memoria a cui assegna nomi univoci, tali aree sono dette variabili.

Interattività

Questo programma sarebbe più utile se permettesse all'utente di specificare i valori di `base` e `altezza` di volta in volta.

Bisogna usare comandi che permettono di introdurre dati dalla tastiera.

AreaRettangoloInterattivo.java

```
import prog.io.ConsoleOutputManager;
import prog.io.ConsoleInputManager;

public class AreaRettangoloInterattivo {
    public static void main(String args[]) {
        ConsoleOutputManager video = new ConsoleOutputManager();
        ConsoleInputManager tastiera = new ConsoleInputManager();
        int base=tastiera.readInt();
        int altezza=tastiera.readInt();
        video.print("L'area è: ");
        video.println(base*altezza);
    }
}
```

Compiliamolo ed Eseguiamolo:

```
[22:20]cazzola@ulik:~/AreaRettangoloInterattivo>javac AreaRettangoloInterattivo.java
[22:20]cazzola@ulik:~/AreaRettangoloInterattivo>java AreaRettangoloInterattivo
5
6
L'area è: 30
[22:25]cazzola@ulik:~/AreaRettangoloInterattivo>
```

AreaRettangoloInterattivo.java: Alcuni Dettagli

`tastiera.readInt()`

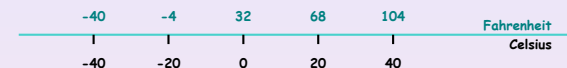
È un comando che:

- attende che l'utente immetta un valore intero;
- fornisce tale valore al programma.

È la controparte di `video.println()`;

`readInt()` accetta una stringa come parametro che viene visualizzata prima di attendere l'input, se omessa non si visualizza nulla.

Celsius vs Fahrenheit



Celsius → Fahrenheit

$$^{\circ}\mathcal{F} = ^{\circ}\mathcal{C} * \frac{9}{5} + 32$$

Fahrenheit → Celsius

$$^{\circ}\mathcal{C} = (^{\circ}\mathcal{F} - 32) * \frac{5}{9}$$

ConvertiTemperature.java

```
public class ConvertiTemperature {
    public static void main(String args[]) {
        ConsoleOutputManager video = new ConsoleOutputManager();
        ConsoleInputManager tastiera = new ConsoleInputManager();
        int celsius; int fahrenheit;
        celsius = tastiera.readInt("Inserisci la temperatura in gradi centigradi");
        fahrenheit = celsius * 9/5 + 32;
        video.print("La corrispondente temperatura in scala fahrenheit è di: ");
        video.println(fahrenheit+" gradi.");
    }
}
```

Compiliamolo ed Eseguiamolo:

```
[00:20]cazzola@ulik:~/ConvertiTemperature>javac ConvertiTemperature.java
[00:20]cazzola@ulik:~/ConvertiTemperature>java ConvertiTemperature
Inserisci la temperatura in gradi centigradi
0
La corrispondente temperatura in scala fahrenheit è di 32 gradi.
[00:25]cazzola@ulik:~/ConvertiTemperature>java ConvertiTemperature
Inserisci la temperatura in gradi centigradi
37
La corrispondente temperatura in scala fahrenheit è di 98 gradi.
[00:29]cazzola@ulik:~/ConvertiTemperature>
```

Esercizi

AreaRettangoloAbbellito

Abbellire il programma AreaRettangoloInterattivo in modo che segnali che input sta attendendo.

Cosa succede se durante l'esecuzione si AreaRettangoloInterattivo l'utente immette dei valori che non sono numerici? E se immette dei valori non interi?

AltezzaRettangolo

Scrivere un programma che accetta in input l'area e la base di un rettangolo e ne calcola l'altezza.

Che succede se immettiamo 0 come valore per la base?

ConvertiChilometriInMiglia

Scrivere un programma che realizzi la conversione tra chilometri e miglia, tenuto conto il fatto che un miglio equivale a 1.61 Km.