Corso di JAVA by E.M.

enrico-marinoni@libero.it

Indice generale

INTRODUZIONE

Java è un linguaggio ad **OGGETTi** (***OOP***) che è indipendente dalla piattaforma HW su cui lo si utilizza e per la cui **esecuzione** richiede la **JRE** (Java Runtime Environment) installata sul sistema su cui si sta operando.
ATTENZIONE:
JRE viene anche chiamata **JVM** (Java Virtual Mchine) che in modo "tecnico" viene anche chiamata **J2SE.**

Per lo **sviluppo** di JAVA è richiesto il **JDK** (Java Development Kit) **+ JRE** e i programmi possono essere sviluppati usando un editor di linea e poi compilando il tutto usando il comando
javac *nomefile.java*
per eseguire il programma si usa invece
java *nomefile*

Esistono anche ambinti integrati detti anche **IDE** (Itegrated Development Enviroment) che facilitano la stesura e il debug dei programmi, i più noti sono:
NetBeans supportato direttamente da SUN e quindi da preferire, la cui doc si trova su
http://www.netbeans.org/
oppure direttamente su
http://java.sun.com/
Eclipse è un'altro ottimo ambiente di sviluppo che supportata anche altri linguacci come C, HTML ecc e la cui doc si trova su:
http://www.eclipse.org/

CONFIGURAZIONE dell'Ambiente di Sviluppo

Per prima cosa serve procurarsi il **JDK** per fare ciò basta andare sul sito http://java.sun.com/ e sulla destra, nella sezione **Popular DownLoad**, cliccate su **JavaSE.**
A questo punto basta fare il download dell'ultima versione disponibile del **JDK** che al momento in cui scrivo è **JDK 6 Update 4.**
JDK contiene anche il JRE o JVM.
ATTENZIONE:
Scegliete la piattaforma su cui andrete a sviluppare, attualmente la scelta è possibile tra:
Linux
Solaris
Windows

Installazione di JAVA su Windows-XP

Fatto il download del JDK lanciate l'installazione e alla fine troverete JAVA installato in **c:\Programmi\Java**
Nel caso del **JDK 6 Update 4** vi troverete le due directory sotto riportate.
c:\Programmi\Java\jdk1.6.0_03
c:\Programmi\Java\jre1.6.0_03
Fatta l'installazione occorre sistemare la PATH di sistema in modo tale da poter aprire una finestra DOS e lanciare direttamente i comandi javac e java senza dover ogni volta specificare il percosro.
In WindowsXP basta aprire **Pannello di Controllo** e fare doppio clic sull'cona **Sistema.**
A questo punto cliccate sulla scheda Avanzate e fare clic sul pulsante **Variabili d'Ambiente.**
A questo punto compare una nuova finestra dalla quale si deve **evidenziare la riga PATH** e poi premere **modifica.**
A questo punto compare una nuova finestra dove sulla riga **Valore Variabile** dovete andare in fondo e inserire la scritta sotto:
;C:\Programmi\Java\jdk1.6.0_03\bin;C:\Programmi\Java\jre1.6.0_03\bin
Fatto quanto sopra date **OK** e uscite.
L'installazione è terminata.

Installazione di JAVA su Windows98

In Windows98 occorre modificare la **path** che si trova all'interno del file **autoexec.bat** che si trova in c:
Per fare le modifiche necessarie usiamo un editor qualsiasi e apriamo autoexec.bat
Andiamo alla fine del file e inserimao le due linee sotto riportate:

```
SET CLASSPATH=.;c\programmi\Java\jdk1.6.0_03\lib\classes.zip;
SET PATH=%PATH%c:\programmi\Java\jdk1.6.0_03\bin;
```

Dopo aver fatto le modifiche salviamo il file e riavviamo il PC.

Installazione di JAVA su LINUX XUBUNTU

Scrittura del primo programma in Java x Windows

Se siete arrivati sino a qui è opportuno provare a scrivere un piccolo programma per verificare se l'installazione è andata a buon fine.
Come prima cosa aprite NotePad o un qualsiasi editor e ricopiate il codice sotto riportato:

```
class Hello {
     public static void main(String args[])
          {
          System.out.println(”Hello...”);
          }
}
```

Salvate il file con il nome **Hello.java**

Aprite una finestra **DOS** e per fare ciò sotto Windows-XP fate:
*Selezionare **START** poi **RUN** e nella finestra che compare scrivete **cmd** e date **OK**.*
Dalla finestra DOS portatevi nella directory in cui avete salvato il file Hello.java
date i comandi sotto riportati
javac Hello.java
java Hello
se tutto è andato bene nella finestra DOS vi deve essere comparsa la scritta **Hello...**

Scrittura del primo programma in Java x LINUX XUBUNTU

Se siete arrivati sino a qui è opportuno provare a scrivere un piccolo programma per verificare se l'installazione è andata a buon fine.
Come prima cosa aprite NotePad o un qualsiasi editor e ricopiate il codice sotto riportato:

```
class Hello {
     public static void main(String args[])
          {
          System.out.println(”Hello...”);
          }
}
```

Salvate il file con il nome **Hello.java**

Aprite una finestra **KONSOLLE** e per fare ciò:
*Selezionare **APPLICAZIONI** poi **SISTEMA** e **KONSOLLE**.*
Dalla finestra che compare portatevi nella directory in cui avete salvato il file Hello.java
date i comandi sotto riportati
javac Hello.java
java Hello
se tutto è andato bene nella finestra DOS vi deve essere comparsa la scritta **Hello...**

CREAZIONE di CLASSI

La sintassi base per dichiarare una ***classe*** è:

```
class NuovaClasse {
     ...
}
```

Se si crea una ***classe*** che eredita delle proprietà da una ***superclasse*** allora la dichiarazione diventa:

```
class NuovaClasse extends NomeSuperClasse{
     ...
}
```

Se la nuova classe che si crea ***implementa un'interfaccia*** si deve specificare il nome dell'interfaccia usando la clausola ***implements*** come riportato sotto:

```
class NuovaClasse implements NomeInterfaccia{
     ...
}
```

Le dichiarazioni di classe che abbiamo visto sopra si possono specificare meglio definendo un ***modo di accesso*** per rendere la classe stessa pubblica, privata ecc.
I modificatori d'accesso disponibili sono:

public
private
protected
abstract
static
final
strictfp

All'interno della classe si possono definire le ***variabili di classe o istanza.***
Di seguito è riportato un'esempio.

```
public class Ghitarra extends StrumentiMusicali {
     String tipo;
     String marca;
     String colore;
     int numCorde;

     ...
}
```

Per rendere le ***variabili*** di classe ***visibili*** a tutte le ***istanze*** della classe si usa la parola chiave **static** per cui l'esempio sopra diviene:

```
public class Ghitarra extends StrumentiMusicali {
     static String tipo;
     static String marca;
     static String colore;
     static int numCorde;

     ...
}
```

Si possono dichiarare anche delle ***costanti*** usando la parola chiave ***final*** come negli esempi sotto.

```
final int distanza = 5;
final String msg = “Hello”;
```

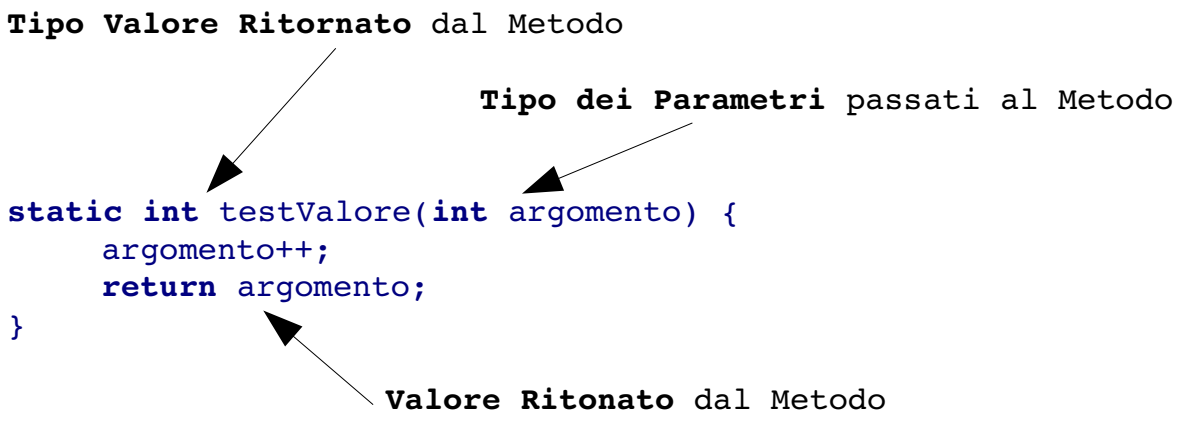
CREAZIONE delle ISTANZE o OGGETTI

Dopo aver creato la classe si possono creare le sue ***istanze*** ovvero gli ***oggetti*** che la compongono.
Per creare le istanze si usa il comando ***new*** come sotto riportato.

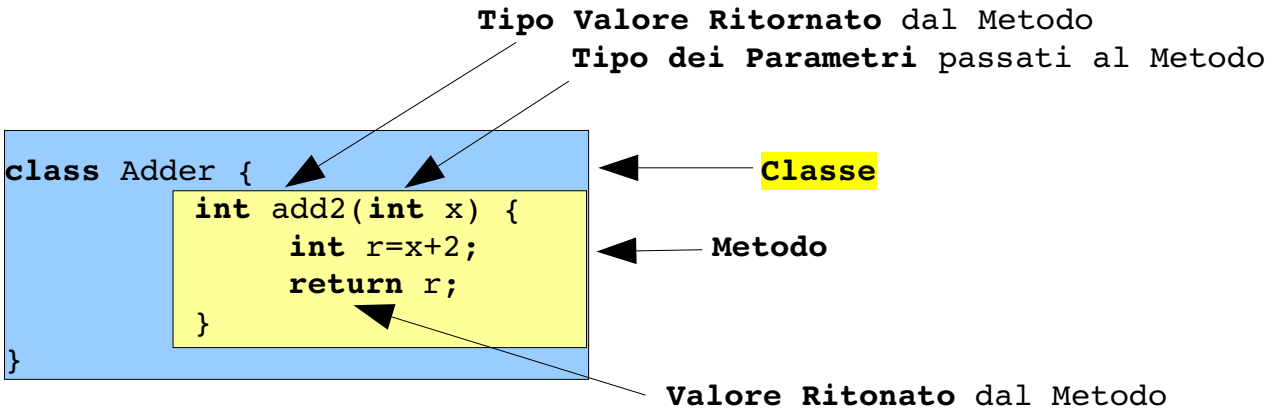
```
NomeOggetto/Istanza = new NomeClasse(parametri);
```

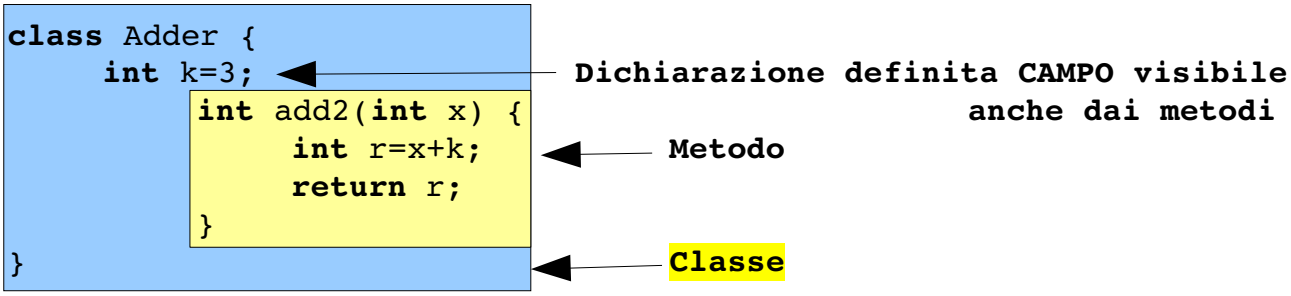
METODI

I metodi sono similari alle funzioni in C e specificano il comportamento degli oggetti di una classe.
La struttura tipica dei metodi è:



In generale le dichiarazione dei **METODI** si devono sempre trovare all'interno di una **CLASSE**
I **nomi** delle **CLASSI devo iniziare** con **Lettere Maiuscole**





COMMENTI

```
// commento su singola riga

/* commento
   multi
   riga */

/** commento per creare
    della DOC Java
*/
```

ARRAY

La definizione tipica di un **ARRAY** è la seguente

```
String[] indirizzi = new String[10];
```

la stessa cosa la si può scrivere anche come sotto:

```
String indirizzi[] = new String[10];
```

Così come si può scrivere anche come sotto:

```
String indirizzi[];
indirizzi = new String[10];
```

Con le istruzioni sopra si crea un **ARRAY** denominato **indirizzi** di tipo **String** composto da **10** elementi.
La **numerazione** all'interno **dell'array** inizia da **0** e nel caso sopra, riferito a **indirizzi**, finisce con **9.**
ATTENZIONE: l'ultima posizione dell'ARRAY contiene il carattere di EOFA per cui, nel nostro caso, gli indirizzi validi saranno da **0** a **9.**

Qui sotto vi è un'esempio su come definire un' ARRAY di numeri

```
int[] arrayOfInts = { 32, 87, 3, 589, 12, 1076,
                              2000, 8, 622, 127 };
```

Ogni elemento di un ARRAY è identificato da un'indice che ne indica la posizione all'interno dell'ARRAY.

Tutti gli ARRAY hanno una **variabile instanza** denominata **length** che ci indica la loro lunghezza ovvero il numero di elementi che compongono l'ARRAY.

Java non permette di definire **ARRAY Multi Dimensionali** veri e propri ma consente di definire di definire ARRAY i cui elementi sono a loro volta ARRAY con risultato analogo.

```
int numeri[] [] = new int[9][9];
numeri[0][0] = 0;
numeri[0][1] = 1;
numeri[1][0] = 0;
...
```

Di seguito è riportato un'esempio di utilizzo degli array.

```
public class Array10 {

   public static void main(String[] args) {
     int n=0;
     String aArray[]= new String[8];
     aArray[0] = "0";
     aArray[1] = "1";
     aArray[2] = "2";
     aArray[3] = "3";
     aArray[4] = "4";
     aArray[5] = "5";
     aArray[6] = "6";
     aArray[7] = "7";

     System.out.println("Array formato da " + aArray.length + "
elementi.");
     for (n=0; n<8; n++){
        System.out.println("Array_"+ n + " contiene... " + aArray[n]);
     }
   }
 }
```

Il risultato a video sarà:

```
Array formato da 8 elementi.
Array_0 contiene... 0
Array_1 contiene... 1
Array_2 contiene... 2
Array_3 contiene... 3
Array_4 contiene... 4
Array_5 contiene... 5
Array_6 contiene... 6
Array_7 contiene... 7
```

ATTENZIONE:
L'esempiop sopra va salvato con il nome di **Array10.java**
Si compila usando il comando javac Array10.java e si esegue usando il comando java Array10

ISTRUZIONI CONDIZIONALI

Istruzione if

Questa istruzione permette di eseguire delle scelte ed è presente in tutti i linguaggi di programmazione.
La sintassi tipica è:

```
if (espressione)
     {
     ...
     }
else if (espressione)
     {
     ...
     }
else if (espressione)
     {
     ...
     }
...
else
     {
     ...
     }
```

Per capire meglio come applicare **if elese** qui sotto è riportato un piccolo esempio.

```
public class If0 {

   public static void main(String[] args) {
     int n=0;
     int x=10;

     if (x==n){
        System.out.println("x e n sono uguali");
     }
     else {
        System.out.println("x e n NON sono uguali");
     }
   }
 }
```

ATTENZIONE:
L'esempiop sopra va salvato con il nome di **If0.java**
Si compila usando il comando **javac If0.java** e si esegue usando il comando **java If0**

Istruzione **switch**

Questa istruzione ci permette di ramificare l'esecuzione del programma in base a possibili valori assunti da una variabile. La sintassi tipica è:

```
switch (condizione) {
     case 1:
          ...
          break;
     case 2:
          ...
          break;
     ...
     default:
          instruzione_di_default;
     }
```

Per capire meglio come applicare **switch** qui sotto è riportato un piccolo esempio che stampa August a video.

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

        int month = 8;
        switch (month) {
            case 1:  System.out.println("January"); break;
            case 2:  System.out.println("February"); break;
            case 3:  System.out.println("March"); break;
            case 4:  System.out.println("April"); break;
            case 5:  System.out.println("May"); break;
            case 6:  System.out.println("June"); break;
            case 7:  System.out.println("July"); break;
            case 8:  System.out.println("August"); break;
            case 9:  System.out.println("September"); break;
            case 10: System.out.println("October"); break;
            case 11: System.out.println("November"); break;
            case 12: System.out.println("December"); break;
            default: System.out.println("Invalid month.");break;
        }
    }
}
```

ATTENZIONE:
L'esempiop sopra va salvato con il nome di **SwitchDemo.java**
Si compila usando il comando **javac SwitchDemo.java** e si esegue usando il comando **java SwitchDemo**

Istruzione **for**

Questa istruzione ci permette di ripetere una o più istruzioni per un numero definito di volte.
La sintassi tipica è:

```
for (inizio; condizione; incremnto o decremnto) {
     ...
     ...
     }
```

Per capire meglio come applicare **for** qui sotto è riportato un piccolo esempio.

```
public class For0 {

   public static void main(String[] args) {
     int n=0;

     for (n=0; n<6; n++){
        System.out.println("n vale: " + n);
     }
   }
 }
```

ATTENZIONE:
L'esempiop sopra va salvato con il nome di **For0.java**
Si compila usando il comando **javac For0.java** e si esegue usando il comando **java For0**

Istruzione **while**

Questa istruzione permette di ripetere una o più istruzioni in base al valore di una condizione specificata.
La sintassi tipica è:

```
while (condizione) {
     ...
     ...
     }
```

Per capire meglio come applicare **while** qui sotto è riportato un piccolo esempio.

```
class WhileDemo {
     public static void main(String[] args){
          int count = 1;
          while (count < 11) {
               System.out.println("Count is: " + count);
               count++;
          }
     }
}
```

ATTENZIONE:
L'esempiop sopra va salvato con il nome di **WhileDemo.java**
Si compila usando il comando **javac WhileDemo.java** e si esegue usando il comando **java WhileDemo**

ATTENZIONE:
Per creare un loop infinito di programma si può usare l'espressione:

```
while(1){
     ...
     ...
     }
```

in questo caso tutto il loop while è sempre verificato per cui NON si interrompe più

Istruzione do while

Questa istruzione permette di ripetere una o più istruzioni in base al valore di una condizione specificata alla fine del loop. In pratica il ciclo **do while** è simile al **while** solo che **while** verifica la condizione prima di eseguire il ciclo.
La sintassi tipica è:

```
do {
     ...
     ...
} while (condizione);
```

Per capire meglio come applicare **do while** qui sotto è riportato un piccolo esempio.

```
class DoWhileDemo {
     public static void main(String[] args){
          int count = 1;
          do {
               System.out.println("Count is: " + count);
               count++;
          } while (count <= 11);
     }
}
```

Come si sarà notato, l'effetto ottenuto, è esattamente lo stesso del programma d'esempio riportato per il while.

ATTENZIONE:
L'esempiop sopra va salvato con il nome di **DoWhileDemo.java**
Si compila usando il comando **javac DoWhileDemo.java** e si esegue usando il comando **java DoWhileDemo**

Istruzione **break**

Questa istruzione può essere usata per interrompere cicli for, while, do while ecc prima del loro naturale termine.
Per capire meglio come applicare **break** qui sotto è riportato un piccolo esempio.
Nell'esempio viene cercato il numero **12** nell'array **arrayOfInts**, quando viene trovato si interrompe il ciclo e viene visualizzata la scritta che è stato trovato il 12 nella posizione 4 dell'array.
Ricordatevi che gli array partono da 0.

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

        int[] arrayOfInts = { 32, 87, 3, 589, 12, 1076,
                              2000, 8, 622, 127 };
        int searchfor = 12;

        int i;
        boolean foundIt = false;

        for (i = 0; i < arrayOfInts.length; i++) {
            if (arrayOfInts[i] == searchfor) {
                foundIt = true;
                break;
            }
        }

        if (foundIt) {
            System.out.println("Found " + searchfor
                               + " at index " + i);
        } else {
            System.out.println(searchfor
                               + " not in the array");
        }
    }
}
```

ATTENZIONE:
L'esempiop sopra va salvato con il nome di **BreakDemo.java**
Si compila usando il comando **javac BreakDemo.java** e si esegue usando il comando **java BreakDemo**

Libreria RxTx

www.rxtx.org

La libreria RxTx ci permette di comunicare con il mondo esterno è free ed è disponibile per Windows, LINUX e Solaris.

A parte questa documentazione altra documentazione su come usarla e installarla la si può reperire sui siti qui sotto riportati.
http://www.javastaff.com/article.php?story=20070423161043906

INSTALLAZIONE

Dopo aver fatto il dowload dei RXTX dobbiamo copiare i file:
RXTXComm.jar in
/home/enrico1/Java/**jre1.6.0_04/lib/ext**
ovvero nella directory in cui abbiamo installato la JRE/LIB/EXT, sopra è riportata la mia installazione.
rxtxserial.dll ed eventualmente **rxtxparallel.dll** in
/home/enrico1/Java/**jre1.6.0_04/bin**