Introduzione alla Programmazione ad Oggetti e allo Sviluppo di Programmi in Java

Unità didattica 1

Armando Tacchella

Fondamenti di Informatica

Oggetti e Classi

- I programmi in linguaggi ad oggetti sono costituiti da un insieme di oggetti che interagiscono in maniera prestabilita per svolgere un determinato compito
- Un oggetto è un'entità caratterizzata da uno stato (le informazioni contenute nell'oggetto) e da un comportamento (le modalità con cui l'oggetto interagisce)
- Prima di utilizzare oggetti in un programma dobbiamo definire le informazioni contenute e il comportamento
- Una classe definisce un insieme di oggetti contenenti lo stesso tipo di informazioni e con lo stesso comportamento

Esempio: conto corrente

- Il conto corrente è una classe
- Ogni conto corrente:
  - Contiene le informazioni relative al cliente, alle condizioni applicate, alle operazioni effettuate, ...
  - Consente di prelevare e depositare
- Il mio conto corrente è un oggetto e contiene le mie informazioni personali, le condizioni che mi sono state praticate, le operazioni che ho effettuato…
- Ogni istituto di credito propone diverse tipologie di conti correnti, quindi diverse classi
- Un istituto di credito ha attivi migliaia di conti correnti, quindi diversi oggetti

Primo programma in Java

- Un programma per mostrare una finestra sullo schermo.
- La dimensione della finestra è leggermente inferiore a quella dello schermo, e la finestra è posizionata al centro dello schermo con il titolo Sample Java Application.
- La classe che definisce gli oggetti finestra si chiama MainWindow ed è contenuta in un package esterno chiamato javabook
- Un package raggruppa un insieme di classi che realizzano funzionalità non previste nel linguaggio base
- javabook contiene classi che consentono di gestire in modo semplice l'IO di dati con interfacce grafiche

PrimoProgramma

```
/*
   Programma PrimoProgramma

   Questo programma mostra una finestra sullo schermo. La
   finestra è al centro dello schermo, e la dimensione della
   finestra è leggermente inferiore a quella dello schermo.
*/

import javabook.*;

class PrimoProgramma
{
   public static void main(String[ ] args)
   {
      MainWindow   finestra;
      finestra = new MainWindow();
      finestra.show()
   }
}
```

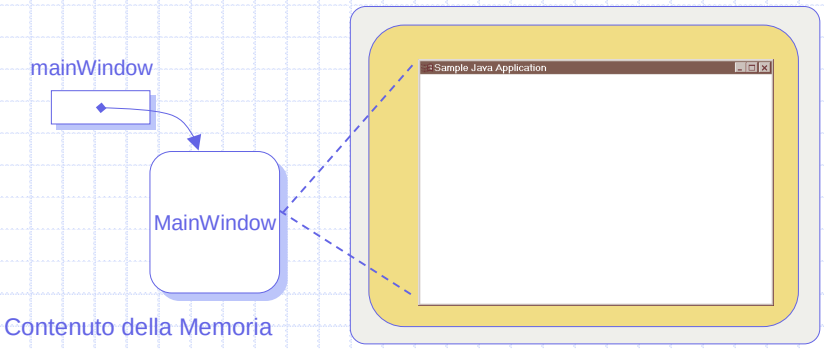
Dichiarazione

Creazione

Interazione

PrimoProgramma in pratica

```
MainWindow  finestra;
finestra = new MainWindow();
finestra.show();
```



Dichiarazione di oggetti

Nome della Classe
La classe deve essere definita prima di questa dichiarazione.

Nome dell'Oggetto
Un solo oggetto viene dichiarato.

```
MainWindow      finestra;
```

Altri Esempi

```
Customer    customer;
Student     jan, jim, jon;
Vehicle     car1, car2;
```

Creazione di oggetti

Nome dell'Oggetto
Nome dell'oggetto che viene creato.

Nome della Classe
Classe di cui stiamo creando l'istanza.

Argomenti
Nessun argomento è necessario in questo caso.

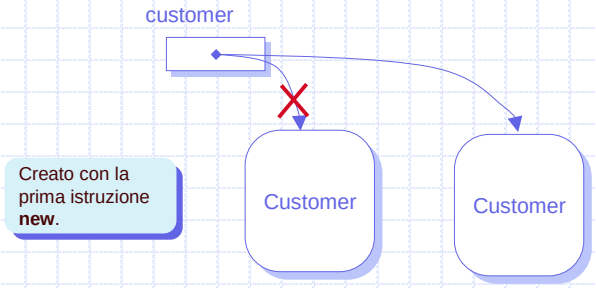
```
finestra  =  new  MainWindow  ( ) ;
```

Altri Esempi

```
customer  = new Customer( );
jon       = new Student("John Java" );
car1      = new Vehicle( );
```

Differenza tra dichiarazione e creazione

```
Customer   customer;
customer   =  new  Customer( );
customer   =  new  Customer( );
```



Creato con la prima istruzione **new**.

Creato con la seconda **new**. Il riferimento al primo oggetto Customer è perso.

Interazione con oggetti

Nome dell'Oggetto
Nome dell'oggetto con cui stiamo interagendo.

Nome del Metodo
Nome del metodo che stiamo chiamando.

Argomento
Nessun argomento è necessario in questo caso.

finestra . show ( ) ;

Altri Esempi

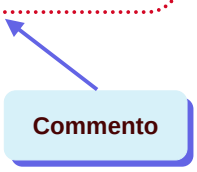
```
account.deposit( 200.0 );
student.setName("john");
car1.startEngine( );
```

Componenti di un programma Java

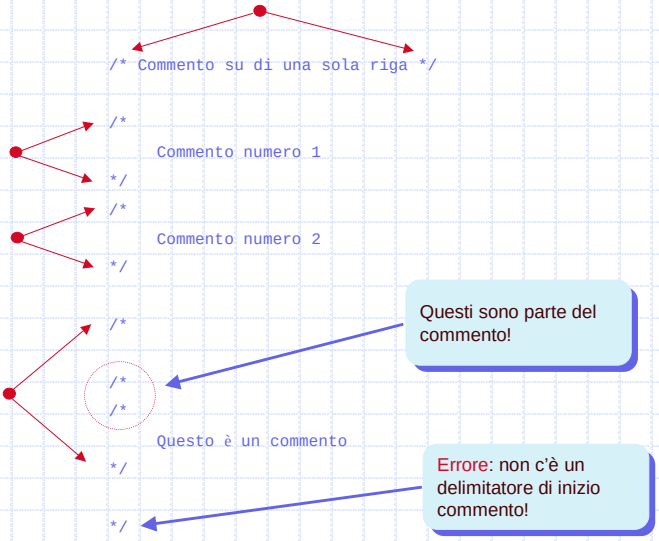
- Un programma è composto da
  - commenti,
  - direttive import, e
  - definizioni di classi.

Componenti: Commenti

```
/*
   Programma PrimoProgramma

   Questo programma mostra una finestra sullo schermo. La
   finestra è al centro dello schermo, e la dimensione della
   finestra è leggermente inferiore a quella dello schermo.

*/
import javabook.*;

class PrimoProgramma
{
   public static void main(String[ ] args)
   {
      MainWindow finestra;
      finestra = new MainWindow();
      finestra.show();
   }
}
```



Delimitazione dei commenti



Tre tipi di commenti

```
/*
     Un commento disposto
     su tre linee di
     testo.
*/
```

Commento su più linee

```
// Un commento
// Un altro commento
// Un terzo commento
```

Commenti su linea singola

```
/**
 * Questa classe fornisce le funzioni basilari di orologio.
 * Oltre all'ora corrente e alla data odierna si può
 * usare questa classe come cronometro.
 */
```

Commenti javadoc

Componenti: direttive import

```
/*
   Programma PrimoProgramma

   Questo programma mostra una finestra sullo schermo. La
   finestra è al centro dello schermo, e la dimensione della
   finestra è leggermente inferiore a quella dello schermo.

*/
import javabook.*;
class PrimoProgramma
{
   public static void main(String[ ] args)
   {
      MainWindow finestra;
      finestra = new MainWindow();
      finestra.show();
   }
}
```

Direttiva Import

Sintassi e semantica di import

Nome del Package
Nome del package che definisce le classi necessarie.

Nome della Classe
Nome della classe da importare. L'asterisco corrisponde a tutte le classi.

<package name> . <class name> ;

e.g. javabook . InputBox;

Altri Esempi

```
import    javabook.*;
import    java.awt.image.ColorModel;
import    com.drcaffeine.galapagos.*;
```

Componenti: definizioni delle classi

```
/*
   Programma PrimoProgramma

   Questo programma mostra una finestra sullo schermo. La
   finestra è al centro dello schermo, e la dimensione della
   finestra è leggermente inferiore a quella dello schermo.

*/

import javabook.*;

class PrimoProgramma
{
   public static void main(String[ ] args)
   {
      MainWindow   finestra;
      finestra = new MainWindow();
      finestra.show();
   }
}
```

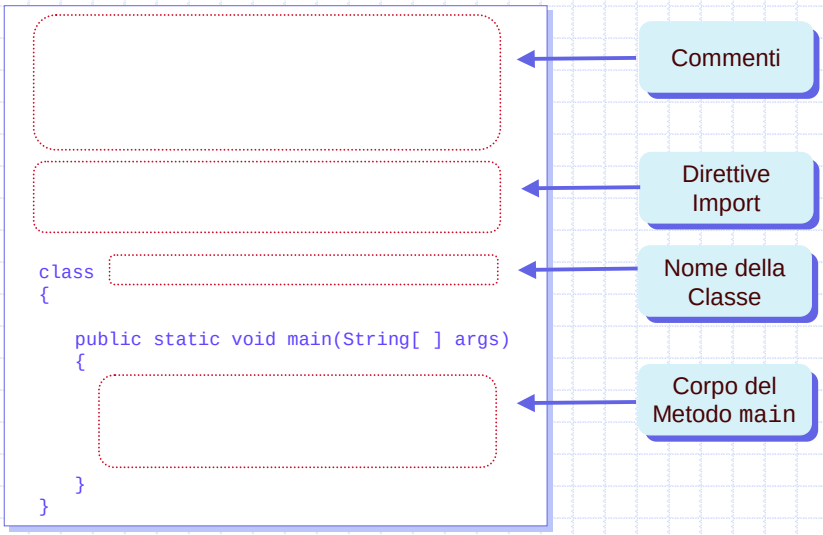
Definizione di una Classe

Definizione di un metodo principale

```
/*
   Programma PrimoProgramma

   Questo programma mostra una finestra sullo schermo. La
   finestra è al centro dello schermo, e la dimensione della
   finestra è leggermente inferiore a quella dello schermo.

*/

import javabook.*;

class PrimoProgramma
{
   public static void main(String[ ] args)
   {
      MainWindow   finestra;
      finestra = new MainWindow();
      finestra.show();
   }
}
```

Definizione di un metodo principale

Schema per semplici programmi Java



Manipolazione di espressioni numeriche

- In Java, l'istruzione per addizionare due quantità x e y è

```
x + y
```

- Tuttavia, prima di effettuare l'addizione è necessario stabilire il tipo delle quantità in gioco. Se x e y sono numeri interi, è necessaria la seguente dichiarazione

```
int x, y;
```

oppure

```
int x;
int y;
```

Variabili

- Al momento della dichiarazione viene riservata una locazione di memoria per contenere i valori x e y.
- x e y sono variabili. Una variabile ha quattro proprietà:
  - un valore
  - una locazione di memoria (che contiene il valore),
  - il tipo di dato contenuto nella locazione di memoria, e
  - il nome usato per riferirsi alla locazione di memoria.
- Esempi di dichiarazione di variabili:

```
int x;
int v, w, y;
```

Tipi per dati numerici

- In Java ci sono sei tipi per dati numerici, i numeri interi byte, short, int, long, e i numeri in virgola mobile float e double.
- Esempi di dichiarazione di variabili numeriche:

```
int     i, j, k;
float   numberOne, numberTwo;
long    bigInteger;
double  bigNumber;
```

- Dichiarando una variabile è possibile inizializzarla. Per esempio si possono inizializzare le variabili intere count ed height, rispettivamente, a 10 e 34 come segue:

```
int    count = 10;
int    height = 34;
```

Precisione e dimensioni dei tipi di dato

Tipo	Contenuto	Valore predefinito	Valore minimo	Valore massimo
`byte`	Intero	0	-128	127
`short`	Intero	0	-32768	-32767
`int`	Intero	0	-2^{31}	$+2^{31} - 1$
`long`	Intero	0	-2^{64}	$+2^{64} - 1$
`float`	Virg. mobile	0.0	-3.40282347 E+38	3.40282347 E+38
`double`	Virg. mobile	0.0	-1.79769313486231570 E+308	-1.79769313486231570 E+308

Istruzioni di assegnamento

- Per assegnare un valore ad una variabile si utilizza un'istruzione di assegnamento.
- La sintassi è la seguente

```
<variabile> = <espressione> ;
```

- Esempi:

```
sum = firstNumber + secondNumber;
avg = (one + two + three) / 3.0;
```

Dichiarazione ed assegnamento

```
int firstNumber, secondNumber;
firstNumber  = 234;
secondNumber = 87;
```

```
A
int firstNumber, secondNumber;
firstNumber  = 234;
secondNumber = 87;
                    B
```

A. Le variabili sono allocate in memoria.

firstNumber 234

secondNumber 87

B. I valori sono assegnati alle variabili.

Codice sorgente **Stato della Memoria**

Assegnamento di dati numerici

```
int number;
number = 237;
number = 35;
```

```
int number;      A
number = 237;    B
number = 35;     C
```

number 35

A. La variabile è allocata in memoria.

B. Il valore 237 è assegnato a number.

C. Il valore 35 sovrascrive il precedente 237.

Codice sorgente **Stato della Memoria**

Assegnamento di oggetti - 1

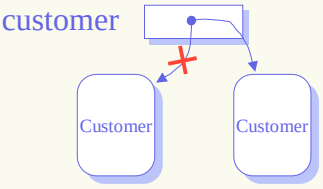
```
Customer customer;
customer = new Customer( );
customer = new Customer( );
```

A

B

```
Customer customer;
customer = new Customer( );
customer = new Customer( );
```

C



A. La variabile è allocata in memoria.

B. Il riferimento al nuovo oggetto è assegnato a customer.

C. Il riferimento al secondo oggetto sovrascrive il contenuto di customer.

Codice sorgente | **Stato della Memoria**

Assegnamento di oggetti - 2

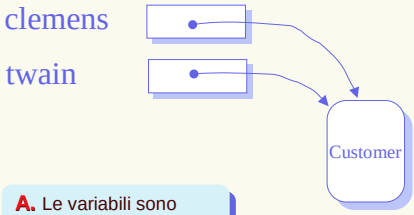
```
Customer clemens, twain;
clemens = new Customer( );
twain   = clemens;
```

A

B

```
Customer clemens, twain;
clemens = new Customer( );
twain   = clemens;
```

C



A. Le variabili sono allocate in memoria.

B. Il riferimento al nuovo oggetto è assegnato a clemens.

C. Il riferimento a clemens è assegnato a customer.

Codice Sorgente | **Stato della Memoria**

Operatori aritmetici

Operazione	Operatore	Esempio	Risultato (x=10, y=7, z=2.5)
Addizione	+	x + y	17
Sottrazione	-	x – y	3
Moltiplicazione	*	x * y	70
Divisione	/	x / y x / z	1 4.0
Modulo	%	x % y	3

In una divisione tra interi la parte frazionaria è troncata.

Espressioni aritmetiche

- Come viene valutata l'espressione seguente?

  x + 3 * y

  Risposta: x è addizionato a 3y.
- L'ordine di valutazione delle espressioni segue rigorosamente delle regole di precedenza.
- Un operatore con precedenza maggiore viene valutato prima di quelli con precedenza minore. Se due operatori hanno la stessa precedenza, la valutazione procede da sinistra a destra.

Regole di precedenza

Priorità	Gruppo	Operatore	Regole
1	sottoespressione	( )	Se le parentesi sono annidate le parentesi più interne sono valutate per prime. Se due o più coppie di parentesi sono sullo stesso livello vengono valutate da sinistra verso destra.
2	operatori unari	+ -	
3	operatori di moltiplicazione	* / %	Se due o più operatori moltiplicativi sono in una espressione vengono valutati da sinistra a destra
4	operatori di addizione	+ -	Gli operatori addittivi sono valutati per ultimi; se due o più operatori additivi compaiono in un'espressione vengono valutati da sinistra a destra.

Conversione di tipo

- Se `x` è un `float` e `y` è un `int`, qual'è il tipo della seguente espressione?

  `x * y`

  Risposta: il tipo risultante è `float`.
- L'espressione sopra è una espressione mista.
- I tipi degli operandi nelle espressioni miste sono convertiti in modo da assicurare che il tipo dell'espressione sarà lo stesso dell'operando il cui tipo richiede la precisione (dimensione) maggiore.

Conversione esplicita di tipo

- Invece di affidarsi alle regole per la conversione implicita è possibile convertire esplicitamente un'espressione da un tipo ad un altro con la seguente istruzione:

```
( <tipo di dato> ) <espressione>
```

- Esempi

```
(float) x / 3
```

Converte **x** in un **float** e poi esegue la divisione.

```
(int) (x / y * 3.0)
```

Converte il risultato dell'espressione **x / y * 3.0** in un **int**.

Ancora sulla conversione di tipo

- Si consideri la seguente espressione:

```
double x = 3 + 5;
```

- Il risultato di 3 + 5 è di tipo `int`. Però, dato che la variabile `x` è un `double`, il valore 8 (tipo `int`) è convertito a 8.0 (tipo double) prima di venire assegnato ad `x`.
- Si noti che il viceversa non è consentito.

```
int x = 3.5;
```

Un valore non può essere assegnato ad una variabile che ha minore precisione.

Costanti

- Il valore di una variabile può cambiare nel tempo. Se un valore rimane invariato, si utilizzano le costanti.

```
final double PI                = 3.14159;
final int    MONTH_IN_YEAR     = 12;
final short  FARADAY_CONSTANT  = 23060;
```

La parola chiave **final** è utilizzata per dichiarare costanti.

Queste sono costanti, dette anche costanti nominative.

Queste sono costanti letterali.

La Classe Math

- La classe Math del package java.lang include molte funzioni matematiche di uso comune, tra cui seno, coseno, tangente, radice quadrata, esponenziale, e altre.
- La formula matematica $\left|\sin \frac{\pi}{4} x\right|$

  è espressa in Java come

```
Math.abs( Math.sin( Math.PI / 4.0) * x )
```

- In particolare `Math.PI` e `Math.E` definiscono, rispettivamente, la costante π e il numero e

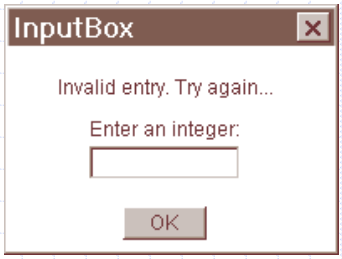
IO con il package javabook: InputBox

- La classe InputBox consente di chiedere all'utente di inserire dati di un certo tipo.

```
InputBox inserisciIntero;
inserisciIntero = new InputBox( finestra );
int x = inserisciIntero.getInteger( );
```

InputBox – Messaggi di errore

- Se viene inserito un valore di tipo inadeguato, si ottiene un messaggio di errore.



InputBox - Personalizzazione

- Il messaggio da visualizzare può essere impostato passando un testo al costruttore come argomento

```
int x = inserisciIntero.getInteger("Enter your age:");
```

InputBox – Alcuni metodi

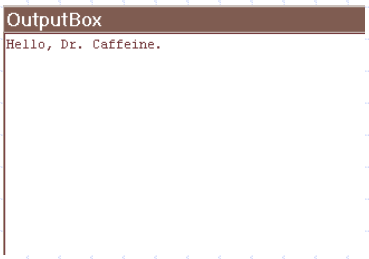
- Tutti i metodi visualizzano un messaggio predefinito, oppure un messaggio fornito dall'utente.

Metodo	Argomenti	Descrizione
`getInteger`	<nessuno> o <testo>	Richiede un numero intero.
`getFloat`	<nessuno> o <testo>	Richiede un numero in virgola mobile.
`getDouble`	<nessuno> o <testo>	Richiede un numero in virgola mobile.

IO con il package javabook: OutputBox

- La classe OutputBox consente di visualizzare dati

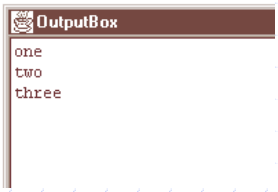
```
OutputBox uscitaDati;
uscitaDati = new OutputBox( finestra );
uscitaDati.print("Hello, Dr. Caffeine" );
```



OutputBox – Metodo printLine

- OutputBox consente di visualizzare più linee di testo utilizzando il metodo printLine.

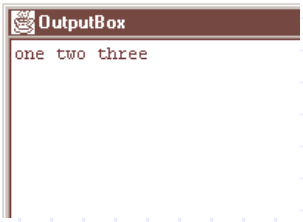
```
uscitaDati.printLine("one" );
uscitaDati.printLine("two" );
uscitaDati.printLine("three" );
```



OutputBox – Metodo print

- A differenza di printLine non inserisce un ritorno a capo tra un testo e l'altro.

```
uscitaDati.print("one " );
uscitaDati.print("two " );
uscitaDati.print("three" );
```



OutputBox – Altri metodi

Metodo	Argomento	Descrizione
skipLine	<numero intero>	Salta N linee, N specificato dall'argomento
saveToFile	<nome file>	Salva il contenuto dell'OutputBox su un file il cui nome è passato come argomento. Se il file esiste già, il suo contenuto è rimosso e rimpiazzato dal contenuto corrente dell'OutputBox
appendToFile	<nome file>	Salva il contenuto dell'OutputBox su un file il cui nome è passato come argomento. Se il file non esiste viene creato, se il file esiste già, il contenuto corrente dell'OutputBox viene aggiunto in coda al file.

Utilizzo di InputBox e OutputBox

- Gli oggetti InputBox e OutputBox sono finestre di dialogo (dialog box) visualizzate sullo schermo
- Una finestra di dialogo deve sempre essere collegata ad una finestra di cornice (frame) che la contiene
- Gli oggetti di classe MainWindow in javabook definiscono finestre di cornice

```
// Creo una nuova cornice
MainWindow finestra = new MainWindow("Cornice");
// Collego inDati a finestra
InputBox inDati = new InputBox(finestra);
// Collego uscitaDati a finestra
OutputBox uscitaDati = new OutputBox(finestra);
```

Esercitazione – Calcolo di aree

- Scrivere un'applicazione Java per il calcolo dell'area di un generico rettangolo
- L'applicazione deve richiedere base e altezza, calcolare l'area e visualizzare il risultato
- La sequenza di passi per la soluzione è:
  - Chiedi la base
  - Chiedi l'altezza
  - calcola area = base * altezza
  - Visualizza l'area
- Modificare il programma per calcolare l'area di altre figure geometriche: triangoli, trapezi, ecc.

Esercitazione – Calcolo di interessi

- Scrivere un'applicazione Java per il calcolo degli interessi semplici e del montante di un investimento
- L'applicazione deve richiedere il capitale iniziale, il tasso di interesse annuale e la durata dell'investimento (anni)
- La sequenza di passi per la soluzione è:
  - Chiedi capitale, tasso e durata
  - Calcola interesse = capitale * tasso * durata / 100
  - Calcola montante = capitale + interesse
  - Visualizza montante e interesse
- Modificare il programma per calcolare la durata dell'investimento dato un capitale iniziale, un interesse e un montante desiderato

Unità didattica 1 – Argomenti svolti

- Definizione di classe e oggetto
- Componenti fondamentali di un programma in Java
- Dichiarazione, creazione e manipolazione di oggetti predefiniti
- Utilizzo della classe MainWindow del package javabook
- Dichiarazione e utilizzo di variabili e costanti numeriche
- Espressioni aritmetiche
- Valutazione di espressioni aritmetiche
- Allocazione di memoria per oggetti e variabili numeriche
- Conversione di tipo
- Utilizzo della classe standard Math
- Utilizzo delle classi InputBox e OutputBox del package javabook
- Sviluppo di programmi interattivi

Appendice – Alcuni metodi della classe Math

Metodi	Descrizione
`sin(x), cos(x), tan(x), asin(x), acos(x), atan(x)`	Funzioni trigonometriche e loro inverse
`toRadians(d)`	Converte la misura d da gradi a radianti
`toDegrees(r)`	Converte la misura r da radianti a gradi
`exp(x), log(x)`	Esponenziale e logaritmo
`pow(y,x)`	y elevato alla x
`sqrt(x)`	Radice quadrata di x
`ceil(x), floor(x), rint(x), round(x)`	Funzioni di arrotondamento
`abs(x)`	Valore assoluto
`max(x,y), min(x,y)`	Massimo e minimo tra due valori