JAVA

OOP

object-oriented programming

Le Classi

Una *classe* è un tipo di dato progettato dal programmatore; essa descrive una specifica categoria di *oggetti* (*istanze*), individuati da comportamenti e caratteristiche simili. I comportamenti vengono definiti *metodi* mentre le caratteristiche (proprietà) vengono definite *attributi*.
Ogni classe è formata da *elementi* o *membri* (*attributi* e *metodi*) i quali vengono definiti all'interno del suo corpo con qualsiasi ordine.

```
public class NomeClasse {
    …
    …
    …
}
```

Il corpo di una classe è formato da un blocco (istruzione composta) che contiene tutta la classe.

Un oggetto è un'istanza (una copia) di una classe; si tratta di variabili che assumono le proprietà specificate dalla classe stessa. Il programmatore può creare (istanziare) più oggetti appartenenti alla stessa classe.
E' importante fare una distinzione tra classe e oggetto; definire una classe significa elencare una serie di proprietà e funzioni (metodi), instanziare un oggetto invece vuol dire creare una variabile che presenta le caratteristiche della classe da cui è stata generata.

La sintassi java per creare un oggetto è la seguente:

NomeClasse nomeOggetto = new NomeClasse(); ➡ Contatore c = **new** Contatore();

Modificatori di accesso

Uno dei principi fondamentali della progettazione delle classi è l'***incapsulamento*** ovvero quel meccanismo che permette di includere al suo interno sia gli attributi sia i metodi che definiscono l'elaborazione di tutti gli oggetti appartenenti alla classe stessa. Tale meccanismo è realizzato mediante i *modificatori di accesso* agli elementi della classe, attraverso i quali è possibile definire quali membri (attributi e metodi) sono accessibili dall'esterno e quali invece sono incapsulati all'interno della classe e quindi non sono visibili al suo esterno. I modificatori d'accesso del java sono:

private: l'elemento dichiarato private è visibile solo all'interno della classe dagli altri membri;

public: l'elemento dichiarato public è visibile sia all'interno che all'esterno della classe;

protected: definisce la visibilità degli elementi all'interno di una gerarchia di classi correlate fra di loro.

Attributi

Gli *attributi* specificano le proprietà (caratteristiche) che tutti gli oggetti di una classe devono possedere. I loro valori, in un certo istante, determinano lo stato di ogni singolo oggetto (istanza) della classe, per questo motivo vengono anche detti ***variabili d'istanza***. Tali attributi vengono dichiarati nel corpo della classe, esternamente ai metodi, pertanto sono visibili e accessibili all'interno di tutti i metodi della classe stessa.

```
public class Contatore {
    private int cont;
        …
        …
}
```

Per realizzare in modo corretto il meccanismo dell'incapsulamento gli attributi di una classe devono essere dichiarati privati, questo impedisce che si possa accedere direttamente ai loro valori dall'esterno.

L'accesso agli attributi privati effettuato dal main, dichiarato all'interno del corpo della classe, è possibile solo in combinazione con un oggetto della classe stessa.

nomeOggetto.nomeVariabile; ➡ c.cont = 10;

Gli attributi dichiarati *private* possono essere manipolati dai metodi esterni, compreso il *main()* dichiarato esternamente alla classe, soltanto attraverso degli appositi metodi pubblici creati dal progettista (interfaccia della classe), i quali devono essere invocati in combinazione con un oggetto della classe stessa.

Variabili locali

Le *variabili locali* sono quelle dichiarate e utilizzate all'interno dei metodi per elaborazioni locali. La visibilità di tali variabili è solo all'interno del metodo in cui sono definite. Al termine dell'esecuzione del metodo, queste variabili vengono deallocate.

Parametri formali e attuali (argomenti)

I parametri sono delle variabili utilizzate per scambiare dati con un metodo, quelli dichiarati nella definizione del metodo sono detti *parametri formali*; invece quelli che vengono passati al metodo all'atto della chiamata sono detti *parametri attuali* (o *argomenti*). Generalmente In Java il passaggio dei parametri avviene per valore, in pratica all'atto dell'invocazione del metodo ad ogni parametro formale viene assegnato il corrispondente argomento. Gli array e gli oggetti invece sono passati per riferimento.

Metodi

I metodi definiscono il comportamento di tutti gli oggetti della classe ovvero tutte le operazioni che possono essere effettuate sugli oggetti. La dichiarazione dei metodi è fatta nel corpo della classe con la seguente sintassi:

modificatoreAccesso tipoValoreRestituito nomeMetodo (lista parametri) {
    …
    …
}

```
public void incremento() {
    cont++;
}
```

Metodo dichiarato *void* che non restituisce alcun valore; inoltre non possiede parametri pertanto le parentesi tonde sono vuote.

All'interno della classe l'accesso all'attributo privato è diretto in quanto esso è visibile in tutto il corpo della classe stessa, ovvero in tutti i metodi definiti al suo interno.

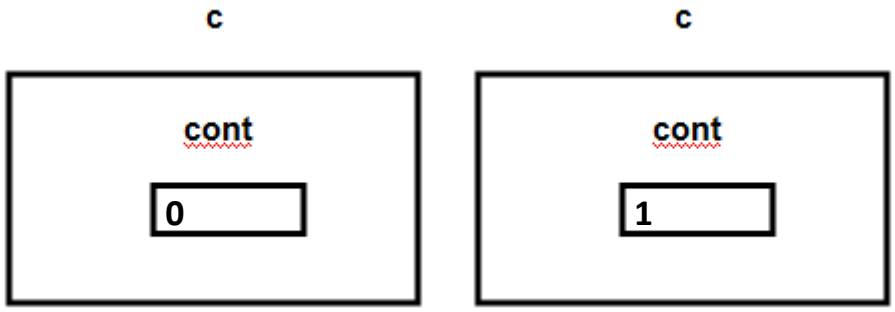
Metodo main

In Java l'esecuzione di un'applicazione stand-alone console comincia sempre da un punto convenzionalmente stabilito, si tratta del metodo *main()*. Tale metodo rende la classe che lo implementa un'applicazione, esso viene richiamato dal sistema operativo per primo all'atto dell'attivazione della stessa. All'interno del main viene inserita la sequenza di istruzioni che realizza l'algoritmo progettato, richiamando altri metodi della stessa classe in cui è definito e/o elementi di altre classi. Esso deve essere definito *public* e *static*, deve accettare un solo argomento, di tipo *String[]* infatti *args* e' un vettore di stringhe che contiene gli eventuali parametri inseriti nel comando di esecuzione, e deve avere *void* come tipo di ritorno.

ESEMPIO 1

```
public class Contatore {
   private int cont;
   public void incremento() {
       cont++;
   }
   public static void main(String[] args) {
      Contatore c = new Contatore();
      c.incremento();
      …
   }
}
```

Il *main()* è pubblico perché deve essere visibile da qualsiasi punto del codice; è *static* perché deve essere invocabile a prescindere dall'esistenza di oggetti; è *void* perché non restituisce alcun valore. Inoltre gli eventuali parametri di input forniti dall'utente sulla riga di comando costituiscono un vettore di stringhe.



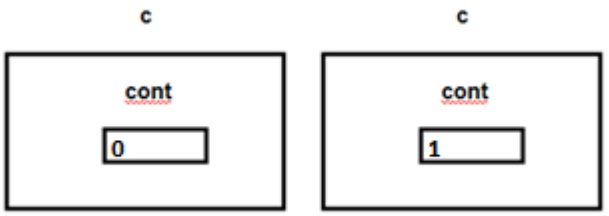
Metodi costruttori

In java il costruttore è il primo metodo di una classe che viene richiamato dopo l'allocazione di un oggetto nella memoria; esso viene invocato esclusivamente con l'operatore *new*.
Il costruttore viene utilizzato per creare e inizializzare un nuovo oggetto, per convenzione ha lo stesso nome della classe e non ha tipo.
Quando il costruttore è definito esplicitamente dal programmatore, può avere dei parametri utilizzati per inizializzare gli attributi dell'oggetto. In assenza di una definizione esplicita sarà il compilatore ad associare automaticamente, a ciascun oggetto, un ***costruttore*** implicito, privo di parametri, cosiddetto di ***default***, che inizializza gli attributi dell'oggetto con valori di default (0, null, false), ciò consente di istanziare correttamente un oggetto anche quando non è stato specificato esplicitamente alcun costruttore da parte del programmatore. Qualora sia stato definito un qualsiasi ***costruttore parametrizzato*** la generazione automatica del costruttore di default viene inibita pertanto, se necessario, deve essere il programmatore a definirne uno.
E' possibile definire più costruttori parametrizzati all'interno della stessa classe, a patto che siano differenziati i parametri formali, nel numero e/o nel tipo (overloading). Il compilatore è in grado di individuare il costruttore da invocare in base al numero e al tipo di parametri attuali passati.

ESEMPIO 2

Nel progetto seguente si riprende l'esempio precedente in cui viene creata una classe con un attributo, un metodo e il *main()*. All'interno del *main()* viene istanziato un oggetto attraverso il quale si invoca il metodo e si accede al valore dell'attributo.

```
public class Contatore {
   private int cont;
   public void  incremento() {
     cont++;
   }
   public static void main(String[] args) {
     Contatore c = new Contatore();
     System.out.println("valore attributo prima dell'incremento " + c.cont); //stampa 0
     c.incremento();
     System.out.println("valore attributo dopo dell'incremento " + c.cont);  //stampa 1
   }
}
```



Questa classe è priva di un costruttore definito dal programmatore, in tal caso interviene il costruttore di ***default*** implicito che azzera l'attributo, infatti alla prima visualizzazione del suo valore avremo zero come risultato.