Classi Java

Le classi estendono il concetto di "struttura" di altri linguaggi

Definiscono

- I dati (detti campi o attributi)
- Le azioni (metodi, comportamenti) che agiscono sui dati

Possono essere definite

- Dal programmatore (ex. Automobile)
- Dall'ambiente Java (ex. String, System, etc.)

La "gestione" di una classe avviene mediante

- Definizione della classe
- Instanziazione di Oggetti della classe

Struttura di una classe

```
public class HelloWorld {

    public static void main(String[] args) {
        System.out.println("Hello World");
    }

}
```

Java è un linguaggio orientato agli oggetti

- In Java quasi tutto è un oggetto
- Come definire classi e oggetti in Java?
- Classe: codice che definisce un tipo concreto di oggetto, con proprietà e comportamenti in un unico file
- Oggetto: istanza, esemplare della classe, entità che dispone di alcune proprietà e comportamenti propri, come gli oggetti della realtà
- In Java quasi tutto è un oggetto, ci sono solo due eccezioni: i tipi di dato semplici (tipi primitivi) e gli array (un oggetto trattato in modo *particolare*)
- Le classi, in quanto tipi di dato strutturati, prevedono usi e regole più complessi rispetto ai tipi semplici

Le classi in Java

- Le classi, in quanto tipi di dato strutturati, prevedono usi e regole più complessi rispetto ai tipi semplici
- Il primo passo per definire una classe in Java è creare un file che deve chiamarsi esattamente come la classe e con estensione .java
- Java permette di definire solo una classe per ogni file
- Una classe in Java è formata da:
- Attributi: (o campi/proprietà) che immagazzinano alcune informazioni sull'oggetto. Definiscono lo stato dell'oggetto
- Costruttore: metodo che si utilizza per inizializzare un oggetto
- Metodi: sono utilizzati per modificare o consultare lo stato di un oggetto. Sono equivalenti alle funzioni o procedure di altri linguaggi di programmazione

---

Incapsulamento e visibilità in Java

- Quando disegniamo un software ci sono **due aspetti** che risultano fondamentali:
  - Interfaccia: definita come gli **elementi che sono visibili dall'esterno**, come il sw può essere utilizzato
  - Implementazione: definita definendo alcuni attributi e scrivendo il codice dei differenti metodi per leggere e/o scrivere gli attributi

---

Incapsulamento

- L'incapsulamento consiste nell'**occultamento degli attributi** di un oggetto in modo che possano essere **manipolati solo attraverso metodi** appositamente implementati. p.es la proprietà `saldo` di un oggetto `conto corrente`
- Bisogna fare in modo che l'interfaccia sia più indipendente possibile dall'implementazione
- In Java l'incapsulamento è strettamente relazionato con la visibilità

Visibilità

- Per indicare la visibilità di un elemento (attribuito o metodo) possiamo farlo precedere da una delle seguenti parole riservate
- `public`: accessibile da qualsiasi classe
- `private`: accessibile solo dalla classe attuale
- `protected`: solo dalla classe attuale, le discendenti e le classi del nostro package

- Se **non indichiamo la visibilità**: sono accessibili **solo dalle classi del nostro package**

Accesso agli attributi della classe

- Gli attributi di una classe sono strettamente relazionati con la sua implementazione.
- Conviene contrassegnarli come `private` e impedirne l'accesso dall'esterno
- In futuro potremo cambiare la rappresentazione interna dell'oggetto senza alterare l'interfaccia
- Quindi non permettiamo di accedere agli attributi!
- per consultarli e modificarli aggiungiamo i metodi accessori e mutatori: `getters` e `setters`

Modifica di rappresentazione interna di una classe

- Uno dei maggiori vantaggi di occultare gli attributi è che in futuro potremo cambiarli senza la necessità di cambiare l'interfaccia
- Un linguaggio di programmazione **ORIENTATO AGLI OGGETTI** fornisce meccanismi per definire nuovi tipi di dato basati sul concetto di classe
- Una classe definisce un insieme di oggetti (conti bancari, dipendenti, automobili, rettangoli, ecc...).
- Un oggetto è una struttura dotata di proprie **variabili** (che rappresentano il suo stato) propri **metodi** (che realizzano le sue funzionalità)

Classi e documentazione

- Come la maggior parte dei linguaggi di programmazione, Java è dotato di una libreria di classi "pronte all'uso" che coprono molte esigenze
- Usare classi già definite da altri è la norma per non sprecare tempo a risolvere problemi già risolti o a reinventare la ruota (DRY)
- La libreria Java standard è accompagnata da documentazione che illustra lo scopo e l'utilizzo di ciascuna classe presente,
- Dalla versione 9 di Java la libreria è stata divisa in moduli
- Documentazione Java 8
- Documentazione Java 9
- Documentazione Java 11
- Documentazione Java 13