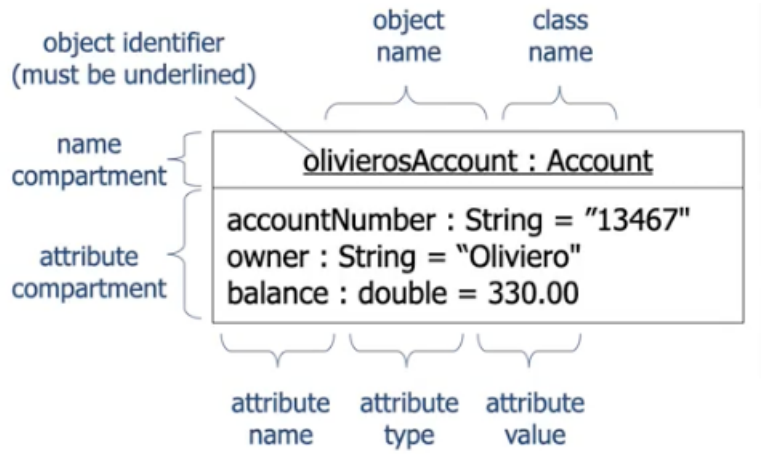
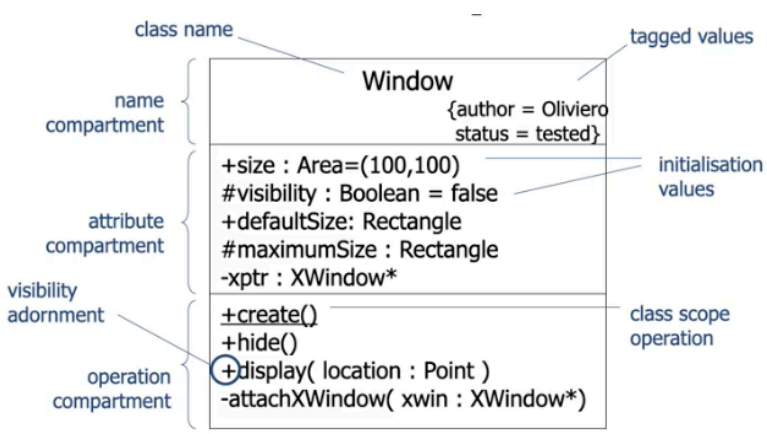
Diagramma iterazioni

Saturday, 1 April 2023 14:46

- Definizione oggetto:
  - Pacchetto coeso di dati incapsulati
  - Ogni oggetto è una istanzia di una classe
    - Valori attributi della classe
    - Operazioni
  - E tutti possiedono:
    - Identità
    - Stato (aka tutti i dati)
    - Comportamento (aka tutte le operazioni)
  - Ogni oggetto è incapsulato (data hiding)
    - Ogni dato è nascosto dentro all'oggetto
    - Per accedere ai dati bisogna chiamare delle operazioni
    - Why? Siccome rende il codice più robusto
  - La collaborazioni tra oggetti avviene tramite messaggi, aka operazioni
  - Seguire le notazioni di queste foto:

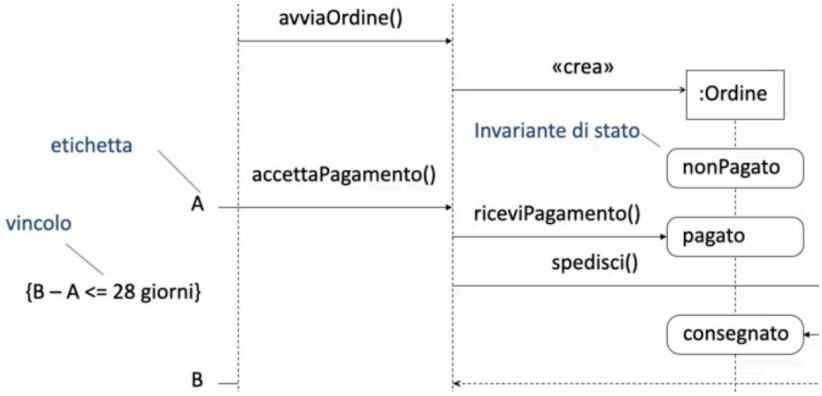




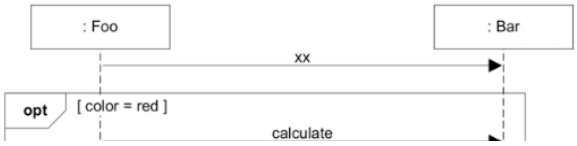
- Iterazione:
  - Come gli oggetti si scambiano informazioni nel tempo
    E qui usiamo le seguenti terminologie:
    - Linee di vita
      - Singolo partecipante ad un istanza
      - Possiedono:
        - Nome
        - Tipo
        - Selettore (condizione che specifica l'istanza)
    - Messaggi
      - Le comunicazioni tra le linee di vita
      - Esistono varie tipologie:

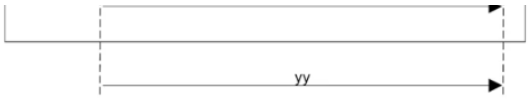
Sintassi	Nome	Semantica
	Messaggio sincrono	Il mittente aspetta che il destinatario ritorni dall'esecuzione del messaggio
	Messaggio asincrono	Il mittente invia il messaggio e continua l'esecuzione: non aspetta un Ritorno dal destinatario
	Messaggio di ritorno	Il destinatario di unmessaggio precedente restituisce il focus di Controllo al mittente di quel messaggio
	Creazione dell'oggetto Distruzione dell'oggetto	Il mittente crea un istanza del classificatore specificato dal destinatario Il mittente distrugge il destinatario. Se la sua linea di vita ha una coda, questa termina con una X
	Messaggio trovato	il messaggio viene inviato dal di fuori dell'ambito dell'interazione. Si vuole mostrare la ricezione del messaggio ma non la provenienza.
	Messaggio perso	Il messaggio non raggiunge mai la sua destinazione. Si può usare per indicare condizioni di errore in cui i messaggi vanno persi.

  - Terminologie in comune per la parte dopo:
    - Stato
      - Condizione/Situazione in cui un oggetto soddisfa condizione/aspetta evento/Attività
      - Vincoli



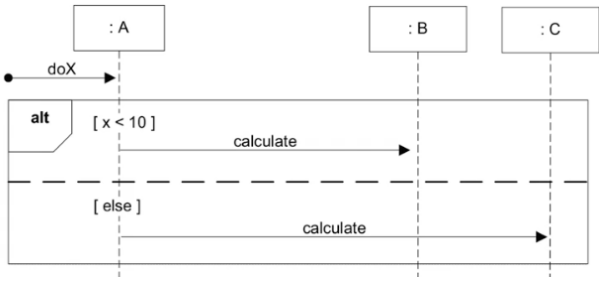
    - Frame combinati
      - Aree in cui abbiamo 1 oppure più operandi
      - Questo andrà a cambiare lo svolgimento
      - Lista operazioni:
        - OPT (if)



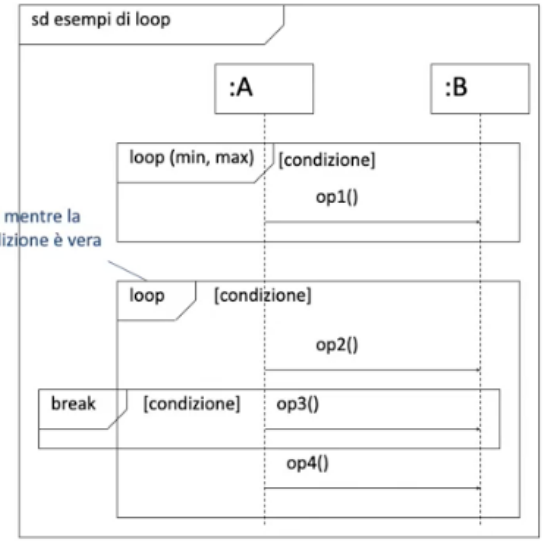


UML 2

- Alt (if else)



- Loop e break



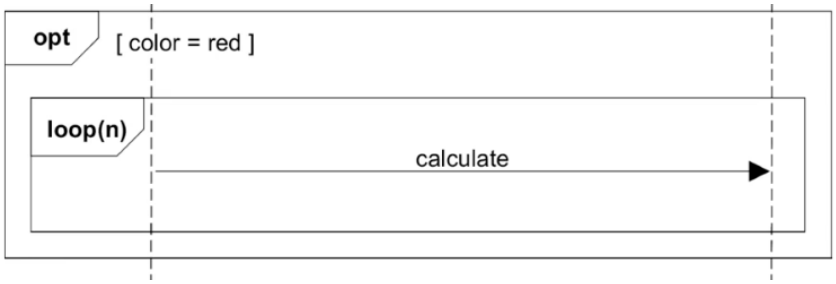
  - Se non si specifica min, min=1
  - Se non si specifica max, loop infinito fino al break
  - Iterazione su collezione:

  - Tipologie di ciclo

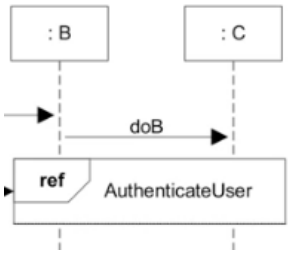
Tipo di ciclo	significato	Espressione del ciclo
While(true) {body}	Continuare il ciclo per sempre	loop *
for i = 1 to n {body}	ripetere n volte	loop n
While(espressioneBooleana) {body}	Ripetere mentre espressioneBooleana è vera	loop [espressioneBooleana]
repeat	Eseguire una volta poi ripetere mentre	loop 1, * [espressioneBooleana]

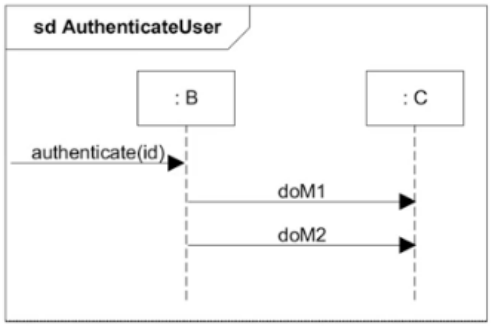
{body} while(espressioneBooleana)	espressioneBooleana è vera	
forEach oggetto della collezione {body}	Eseguire il corpo del ciclo una volta per ogni oggetto di una collezione di oggetti	loop [per ogni oggetto in CollezioneDiOggetti]
forEach oggetto della classe {body}	Eseguire il corpo del ciclo una volta per ogni oggetto di una particolare classe	loop [per ogni oggetto in NomeClasse]

- Annidamenti

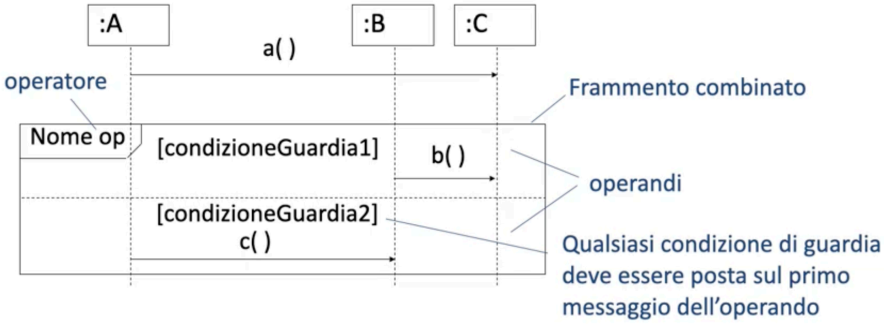


- Ref (funzioni)



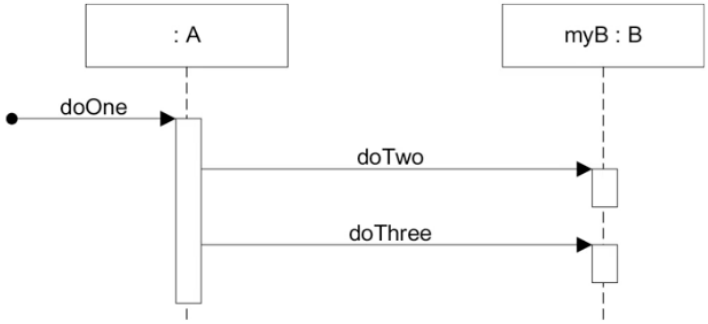


- Condizioni di guardia stabiliscono se gli operandi devono essere eseguiti



- Varie tipologie di iterazioni:
  - Temporizzazione
    - Enfatizza gli aspetti in tempo reale di una iterazione
    - Non lo useremo
  - Sequenza
    - Mostra le iterazioni tra oggetti
    - Si usa un diagramma che va verso sinistra a destra, quindi consuma spazio orizzontale
    - Enfatizza gli scambi temporali

- Mostra iterazioni ordinate
- Non mostra le relazioni tra oggetti, però possiamo dedurli:



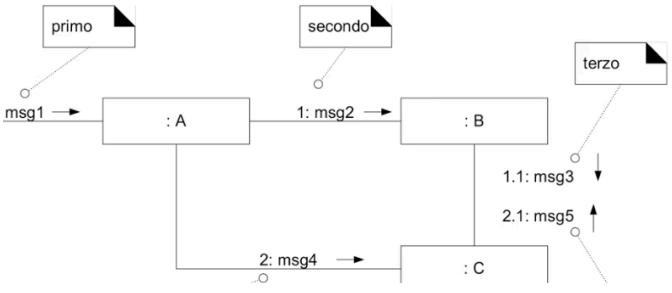
  - In A avremo un metodo doOne che verrà chiamato
  - La classe A avrà un riferimento B
  - La classe B avrà 2 metodi doTwo doThree
  - Sono chiamate sincrone
  - In doOne chiameremo prima doTwo e dopo doThree

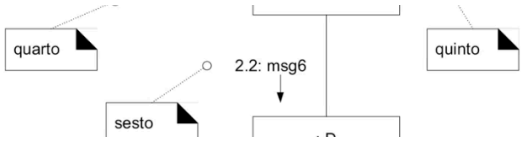
```
public class A {
private B myB = new B();
public void doOne() {
myB.doTwo();
myB.doThree();
}
...
}
```

```
public class B {
public void doTwo() {
...
}
public void doThree()
...
}
...
}
```

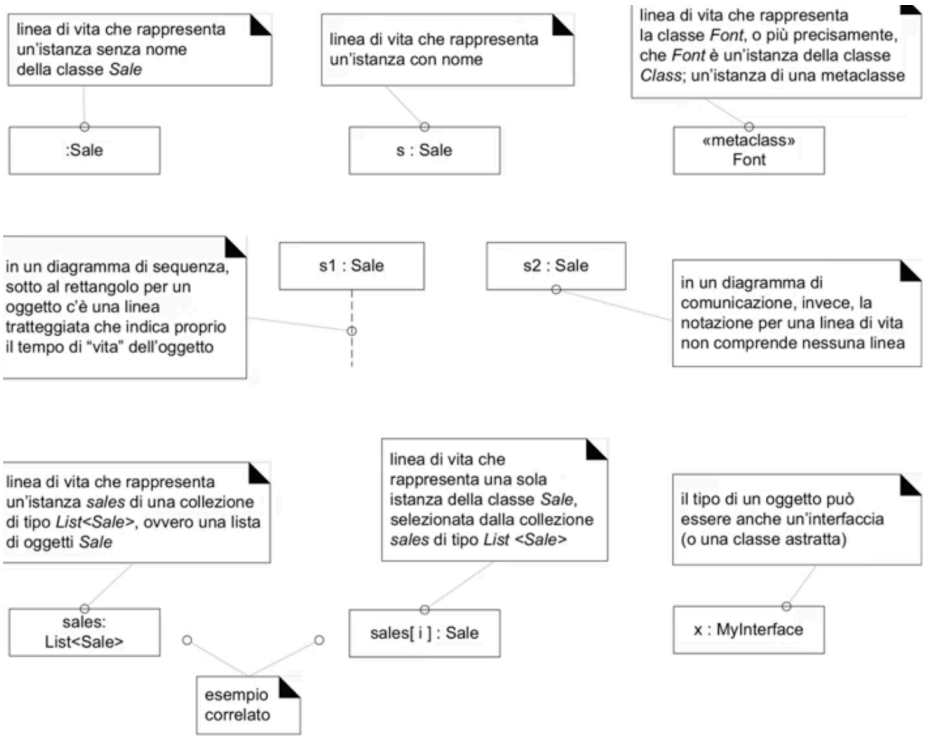
- Comunicazione
  - Iterazione insieme di oggetti
  - Enfatizza relazioni strutturali
  - Esplicita le relazioni tra messaggi

  - Più difficile comprendere la sequenza dei messaggi
  - Elementi:
    - Collegamento
      - Percorso tra 2 oggetti
      - Rappresenta la navigabilità/visibilità
    - Nimerazione messaggi
      - Il primo messaggio non viene mai numerato
      - I messaggi devono comprendere annidamenti
      - I messaggi inviati a se stessi sono chiamati auto-deleghe

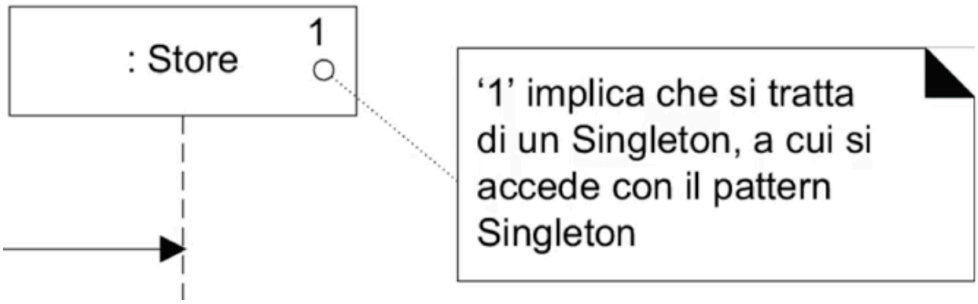




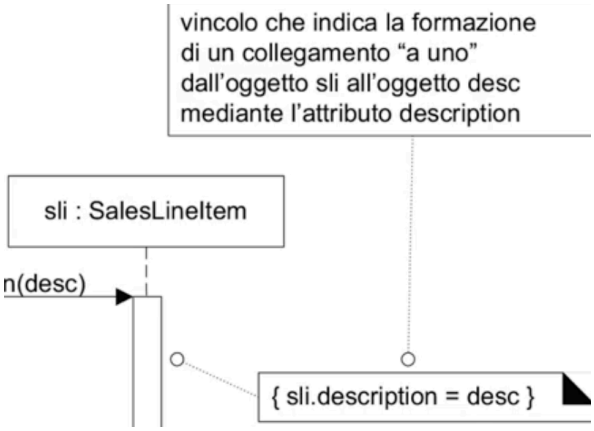
- Notazioni comuni



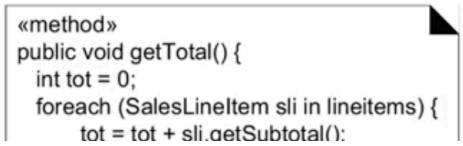
  - Singerton (un oggetto ha 1 ed 1 sola istanza)

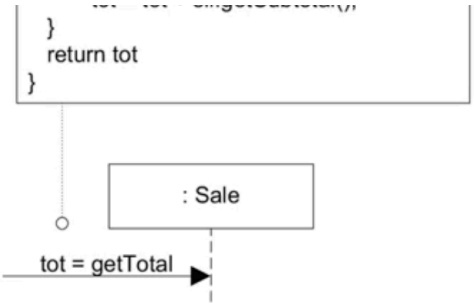


  - Note
    - Esprimere un vincolo



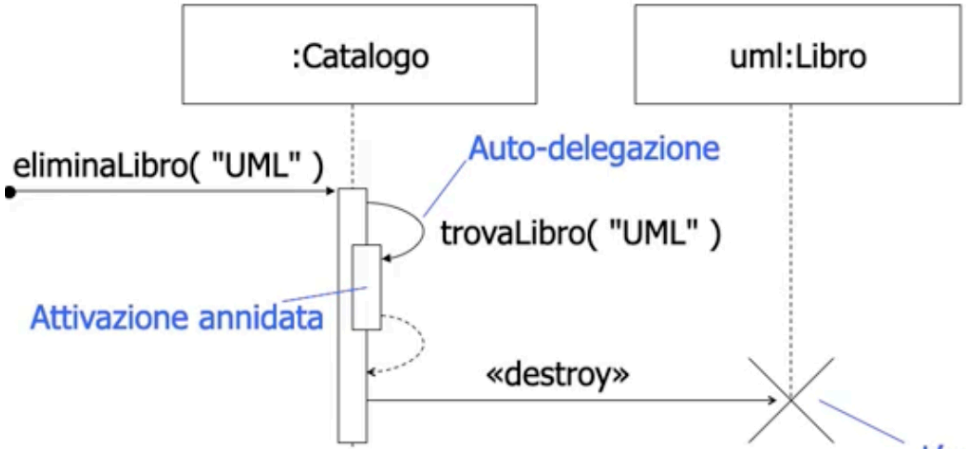
    - Mostrare il corpo di un metodo



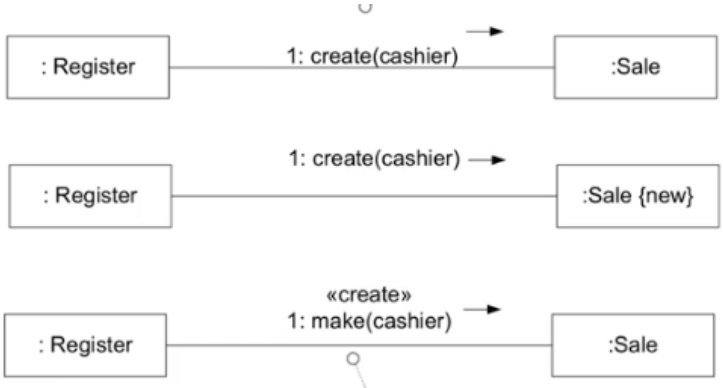


- Valori di ritorno

- Distruzione oggetti con autodelega

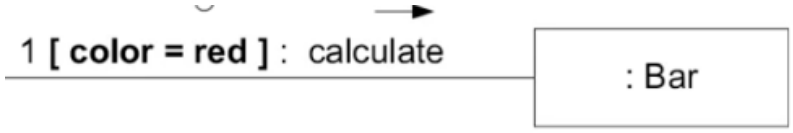


- Creazione di una istanza
  1) Create
  2) (new)
  3) <<create>> → Consigliato

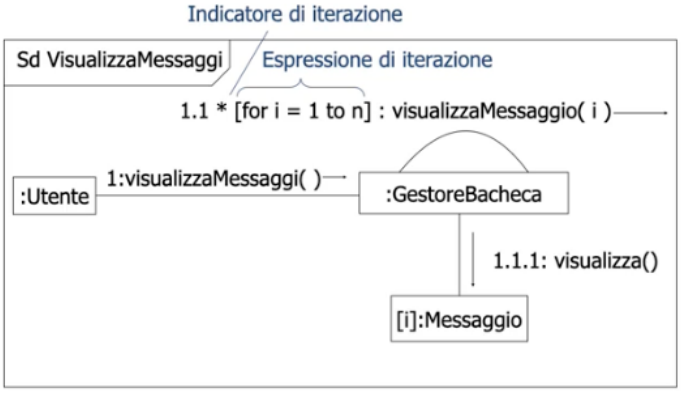


- Associazioni
  - Molti → List
  - Uno → Attributo

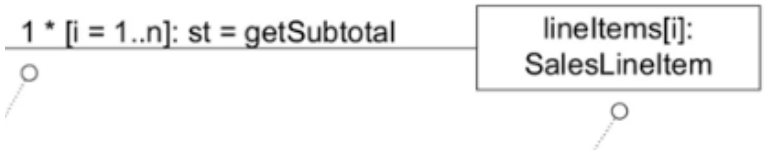
- Operazioni:
  - if



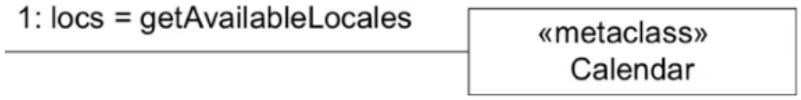
  - Ciclo



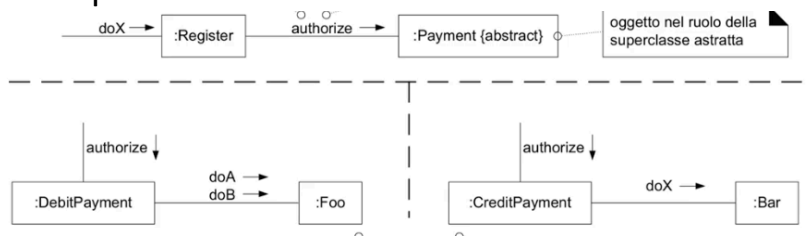
  - Iterazione su collezione



  - Chiamata classi statiche



  - Classi polimorfi



  - Chiamate asincrone