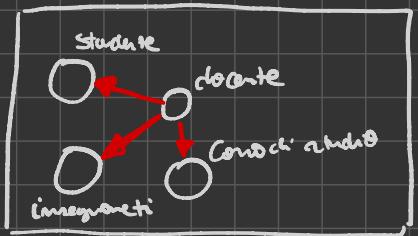


28/03/2023

## ELEMENTI DI INGEGNERIA DEL SOFTWARE → accunato processo di sviluppo

- ↳ ANALISI }
  - ↳ DESIGN }
  - ↳ IMPLEMENTAZIONE / CODIFICA
  - ↳ POST - CODIFICA
- PIÙ DISPENSIONE DI TEMPO

→ RACCOLTA DEI REQUISITI (scrivere tutti i passaggi requisiti funzionali del sistema)  
ELENCO DEI REQUISITI → sia funzionali, che non



Modello del dominio

### PROBLEM SPACE

→ PARADIGMA DI PROGETTAZIONE → ASTRAZIONE → CONCETTO PER RAPPRESENTARE LA SOLUZIONE DEL PROBLEMA

OGGETTO, modello di astrazione

### SOLUTION SPACE

↓  
entità / soluz. pezzi / al fine di risolvere il problema

### PRINCIPI GUIDA :

- DECOMPOSIZIONE → DIVIDI ET IMPERA
- COMPOSIZIONE E RIUSO → LIBERATE RIUSO DI ENTITÀ GIÀ ESISTENTI (MOLTO IMPORTANTE)
- ASTRAZIONE
- SEPARAZIONE

### ASTRAZIONI



### LINGUAGGI DI PROGRAMMAZIONE :

- ↳ LESSICO
- ↳ SINTASSI
- ↳ SEMANTICA

## PARADIGMA DI PROGRAMMAZIONE :

Paradigma imperativo :

- > STRUTTURE
- > PROCEDURE
- > ORIENTATO AD OGGETTI

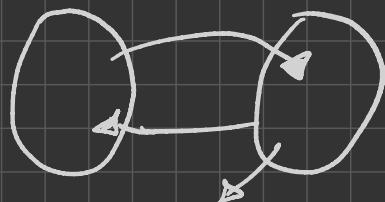
Paradigma funzionale :

Lo side-effect esecuzione delle funzioni può modificare i dati del mio risrone

Programmazione logica:

- > intelligenza artificiale

## PROGRAMMAZIONE AD OGGETTI



simboli -> DATI E INFORMAZIONI

-> ESPOSIZIONE E RELAZIONI

QUALI SONO, COME SONO, COSA CI FACESS

MENU CLASSE

CONTENUTO OGGETTO CHE VIENE USATO PER CONTENERE QUANTITE VOLTE VENNE INTRODOTTO

PRINCIPI FONDAMENTALI (Alan Kay 1995)

- OGNI COSA È UN OGGETTO, CHE SI SCAMBIA MESSAGGI TRA DI LORO E HANNO UNA  
LORO MEMORIA INTERNA

28/09/2023

MODELLO UNA LAMPADINA

isOn();

MODELLO UN BOTTON

- boolean isOn;  
= false;

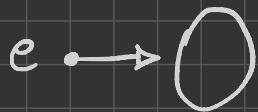


ESPORE FUNZIONALITÀ

Come si deve comporre l'oggetto lampadina  $\rightarrow$  CLASSE Lampadina

```
class Light {           light.joue  
    boolean isOn = False;  
    boolean isOn() { return }  
    void on() { isOn(true) }  
}
```

Light e = new, light()  
 ↑  
 costante



interface Light {

```
bool isOn();  
--- off();  
--- on();
```

}

## PROPRIETÀ

- OGNI OGGETTO HA UN'INTERFACCIA
- INCAPSULAMENTO
- UN OGGETTO DEVE NASCONDERE L'IMPLEMENTAZIONE

JAVA : OGGETTI E CLASSI

04/10/2023

## CLASSE COUNTER

↳ class Counter {

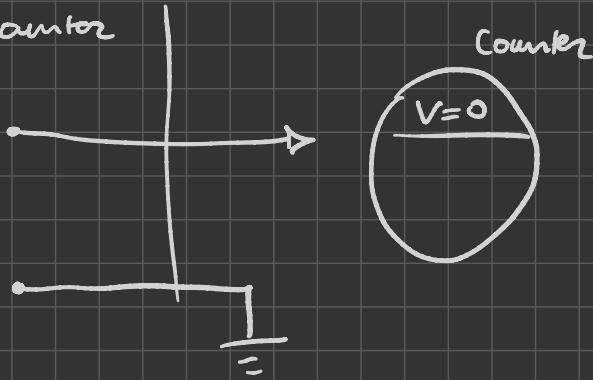
    int v;      // campo delle clone

    void inc() {

        v++;  
    }

}

Use Counter



11/10/2023

ECLIPSE

↳ SRC → SORGENTI  
↳ BIN → .BIN

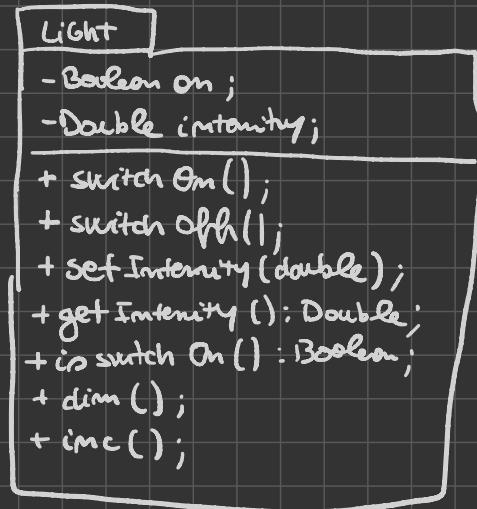
PROGETTAZIONE SISTEMA

① ENTITÀ COINVOLTE

LAMPADINA

BUTTON → CLIENT

② PROB. PARTE PUBBLICA



12/10/2023

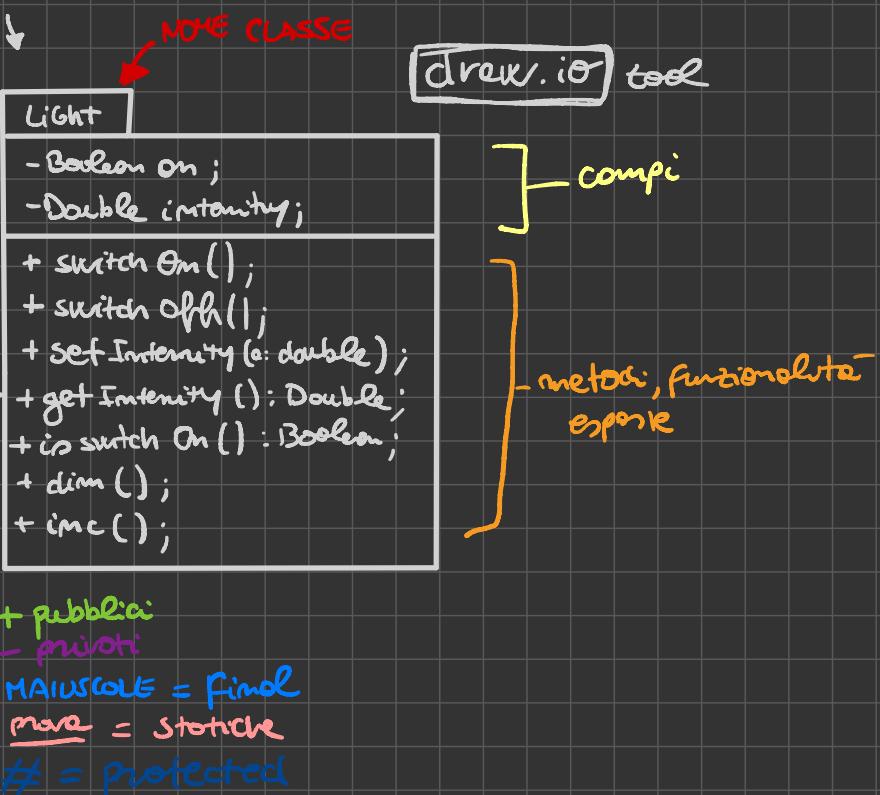
## INCAPSULAMENTO



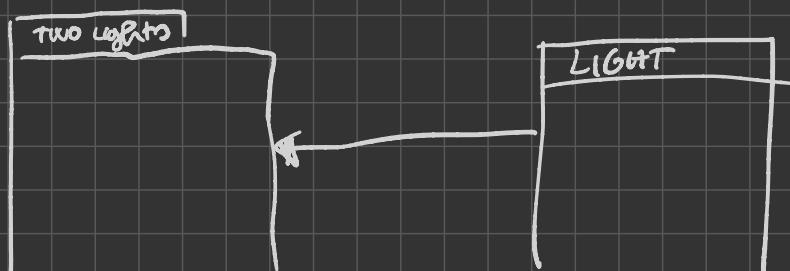
Information hiding

Riuso → **composizione**

UML → standard di riferimento

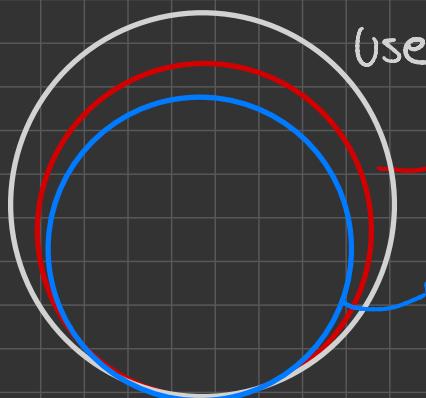


Composizione e utilizzo



riuso cloni nelle classi che principale che mi serve

Composizione

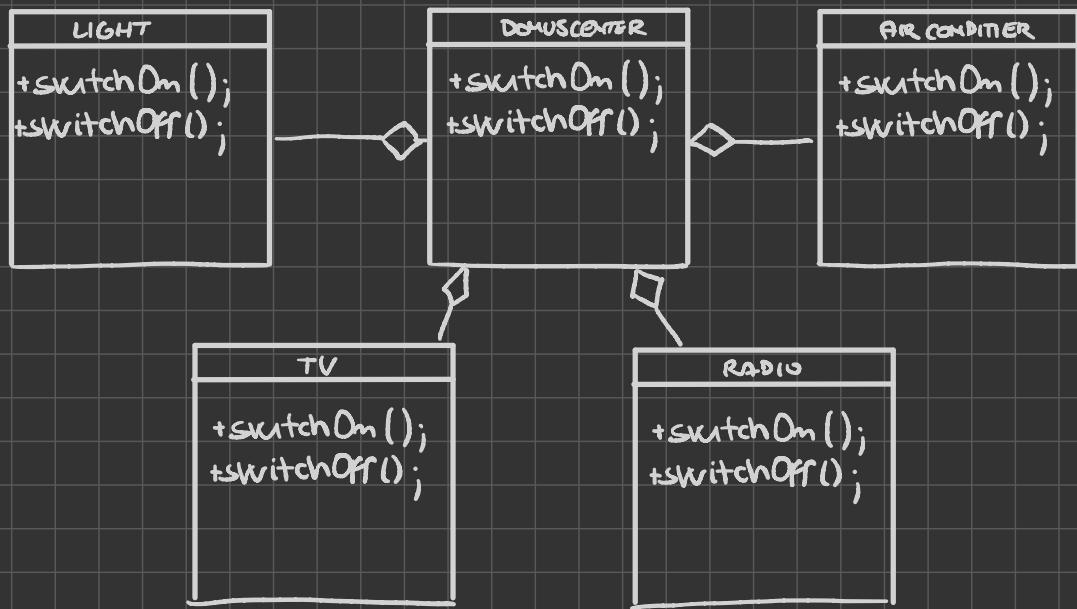


AGGREGAZIONE  
sono più modulari

componibile parte  
non esistono finché non esiste ciò

↓  
figlio → elenco

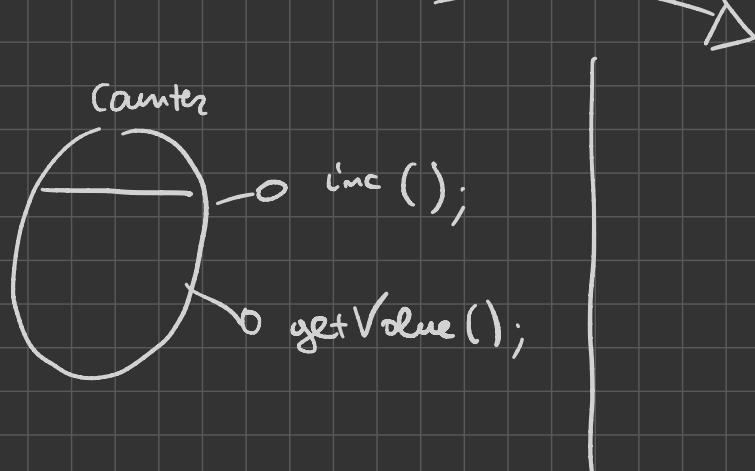
18/10/2023



array di luce:

```
" " TV  
" " Radio
```

## INTERFAZIE



## IMPLEMENTAZIONE

↳ CONTATTATO → INTERFAZIE

Creazione Interface → public interface BonicCounter {

void inc();

int getValue();

↳ non si implementa un  
implementazione  
commune

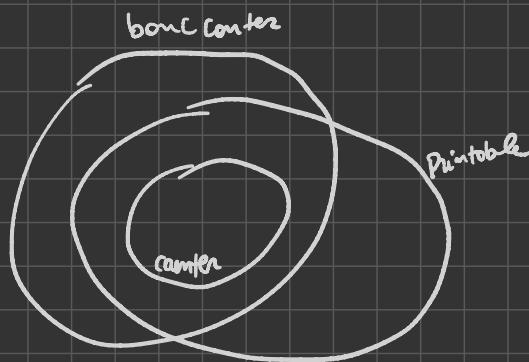
non ha  
stato e default

un tipo di dato

}

o parte riportata ci mette core il contatore se fore  
elle alone

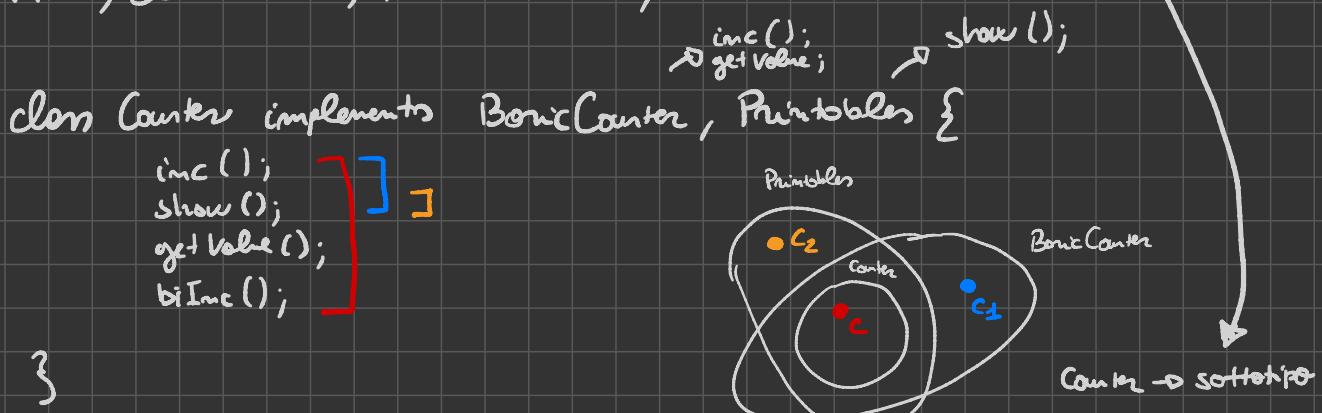
Il dominio di tutti i boor Counter sono tutti gli oggetti che implementano l'interfaccia `BoorCounter`



## PRINCIPIO DI SOSTITUZIONALITÀ

19/10/2023

TIPI, SOTTOTIPI, POLIFORMISMO, SOSTITUZIONALITÀ



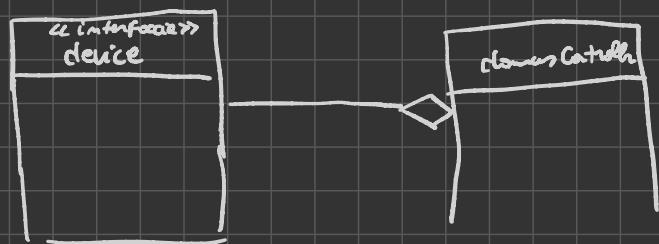
`Printables` → tipo di dato { diverso tra loro

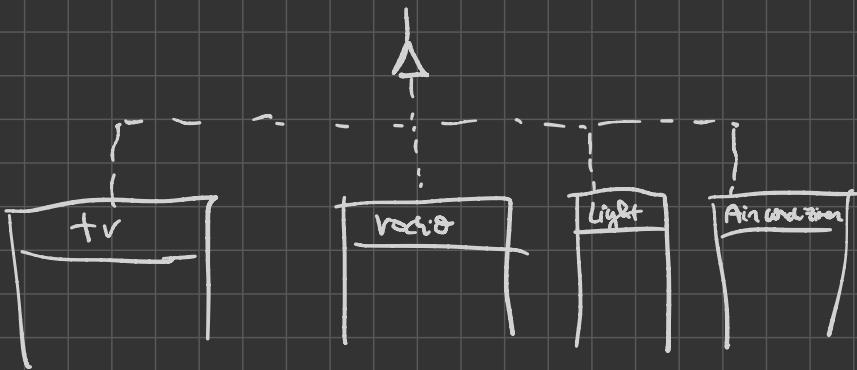
`BoorCounter` → tipo di dato } diverso tra loro

che derivano da un  
Counter

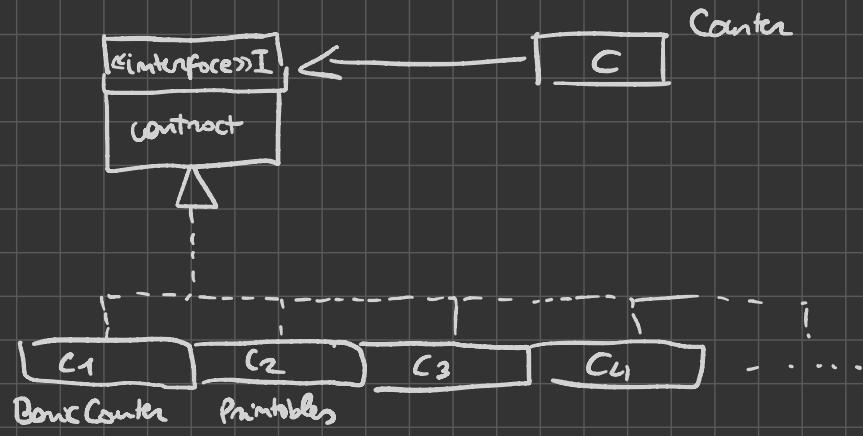
Principio di sostituibilità di Liskov (1993)

$A \rightarrow B$  A si può usare anche se il programma ne ottiene B





## POLIMORFISMO INCLUSIVO

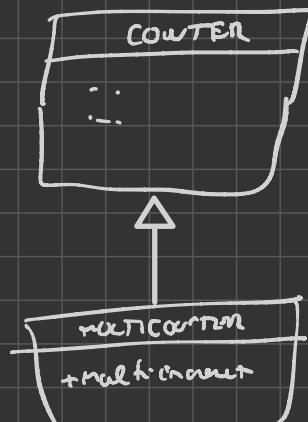


## EREDITÀ RIETÀ

sono un contrattore e lo estendo

```
public class Multicounter extends Counter {  
    ...  
    TUTTO quello che c'è nel counter  
    public void MultiInc (int m) {  
        ...  
    }  
}
```

UML



3

4 livelli

Public

Protected

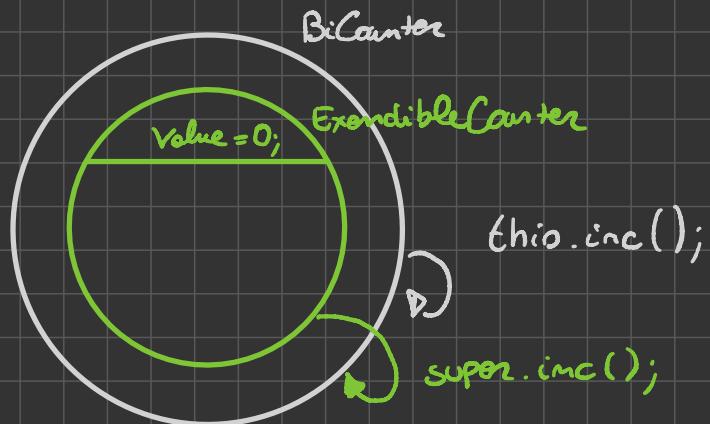
default - package

private

più protetto o  
meno protetto

25/10/2023

Come mi specifico il costruttore da chiamare? → nelle classi base  
↳ tramite super(); → clone parente che mi deve costruire



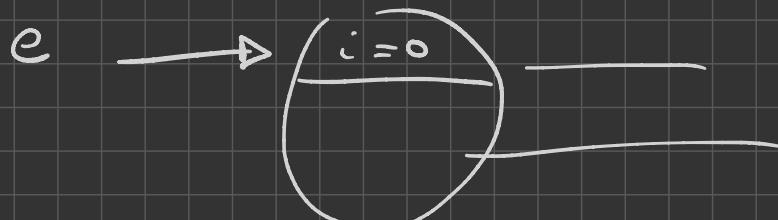
Prima bisogna costruire il core dell'oggetto poi puoi clonare oggetti anche all'esterno

Super. è analogo al this. può essere usato

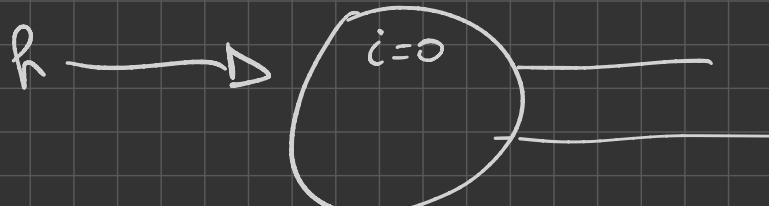
[  
super();  
super.compo();  
super.metodo();]

OVERRIDING DEL METODO

DISPATCH TABLE



NM	NC
m()	E
m()	E

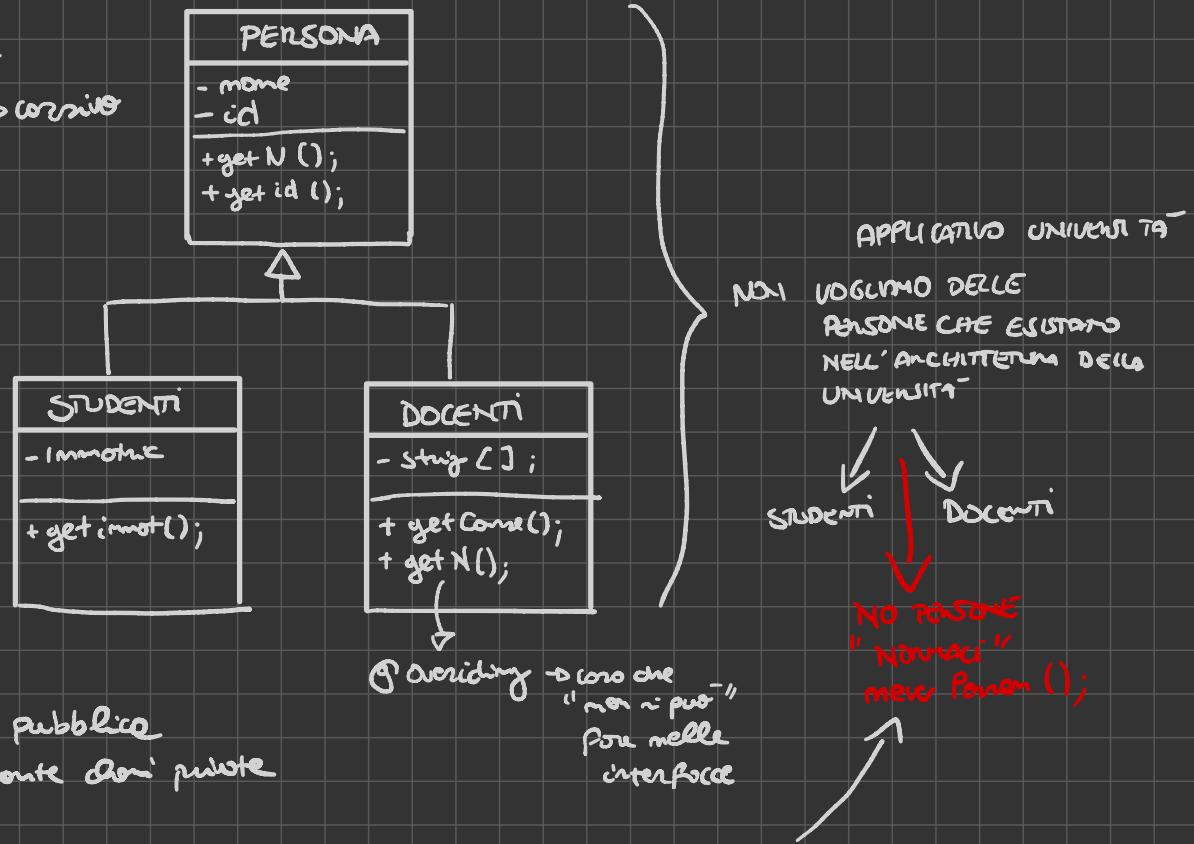


NM	NC
m()	F
m()	E
O()	F

26/10/2023

## MOTIVAZIONI → CLASSI ASTRATTE

metodo: entro nello  
↳ UML → corso

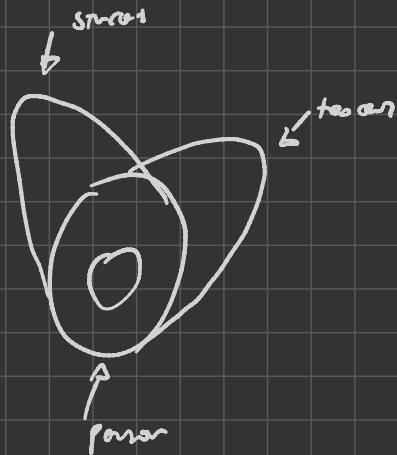


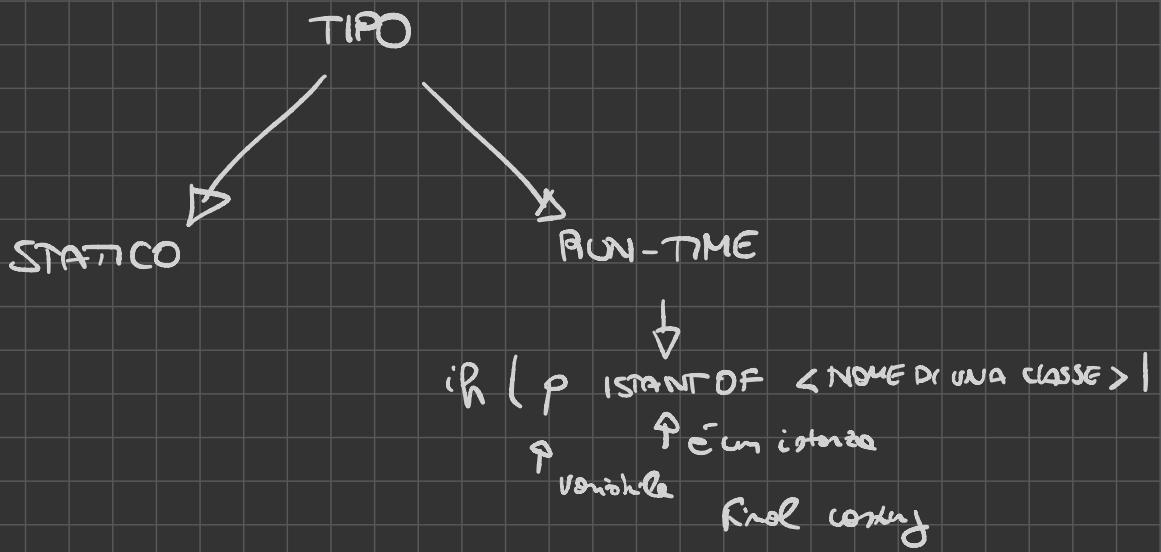
1 file 1 classe pubblica  
↳ tutte classi private

ABSTRACT CLASS → NON POSSO CREARE OGGETTI DELLA CLASSE ASTRATTA

↳ POSSO DEFINIRE RETRAS ASTRATTO  
ABSTRACT VOID prova();

## POLIFORMISMO





02/11/2023

IL POLIFORMISMO PARMETRICO  $\rightarrow$  INTRODOTTO E UTILIZZATO NEL PROBLEMA DELLE COLLEZIONI

STRUCTURE  
DATI

$\hookrightarrow$  COLLEZIONI POLOMORFICHE CHE INDIPENDENTEMENTE DAL TIPO DI DATI CI RESTITUISCE CERTE FUNZIONALITÀ

STRUCTURE DATI: PAIR

$\hookrightarrow$  2 ELEMENTI

GENERALI PARMETRIZZATI SU INTERFACE      ARRAY DI DEVICE PARMETRIZZATI

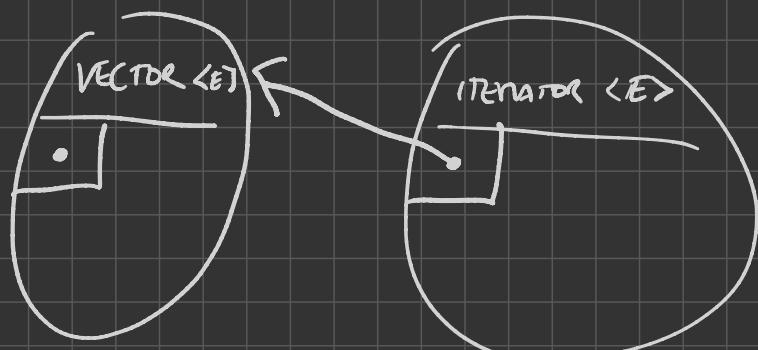
09, pag 23

INTERFACCIA ITERATOR  $\rightarrow$  09.23 SLIDE

$\rightarrow$  next

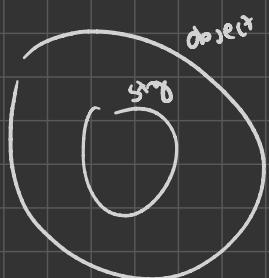
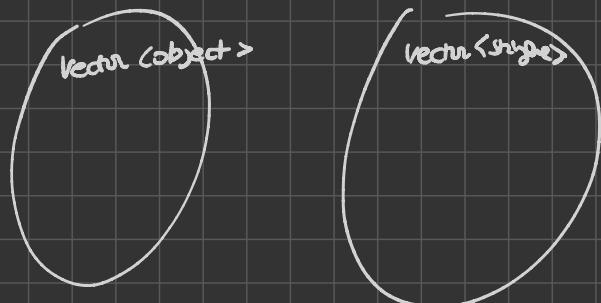
INTERFACCIA

$\hookrightarrow$  [start, end]    start ↑ 5, start ↑ 6, - -



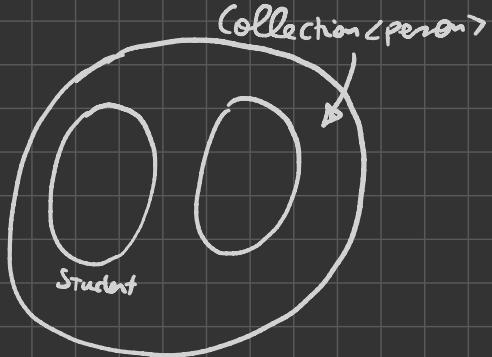


(“ ”)



03 | 11 | 2023

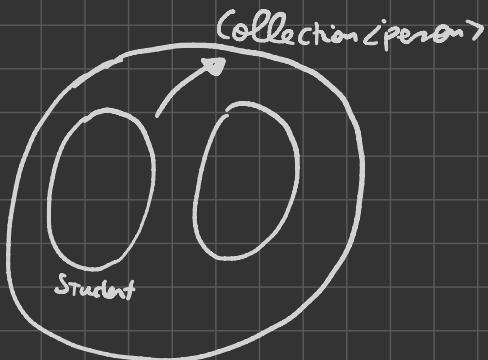
$C < ? \text{ extends } T >$  → extend le filtre



$\text{Collection} < \text{per}$

print ( $\text{collection} < ? \text{ extends } T >$ )  
on

$C < ? \text{ super } T >$  → extend il passe



$\text{Collection} < \text{student} >$

print ( $\text{collection} < ? \text{ super } T >$ )

# JAVA COLLECTION FRAMEWORK (Java dec 17)

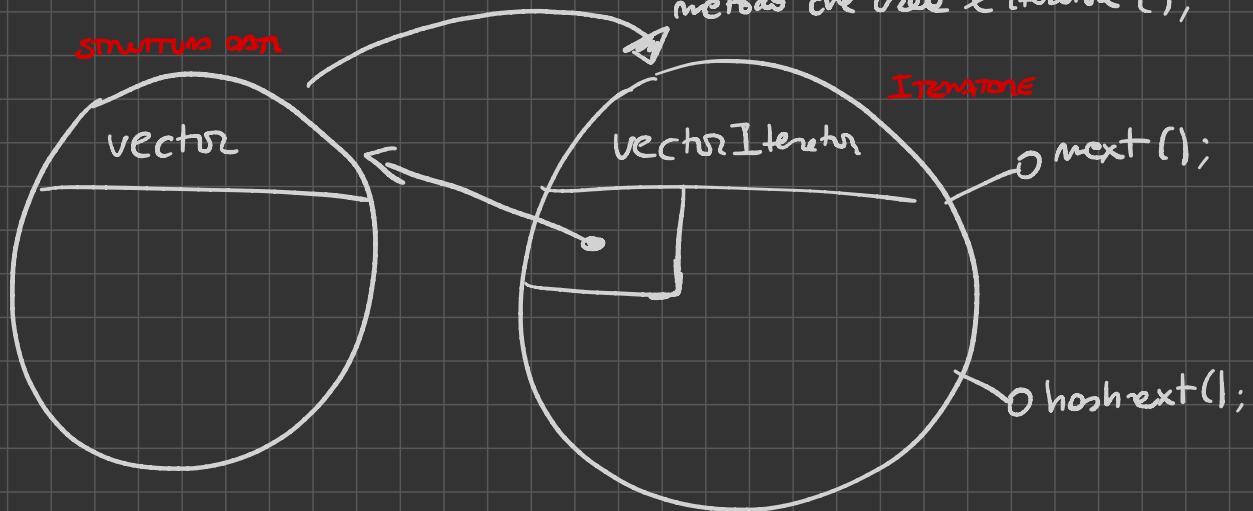
Trotteggiate interfacce

Trotteggiate lungo clone o non

Spendi clone verso e mania

foreach non si può usare direttamente nato cintante di Iterable

metodo che crea l'iteratore();



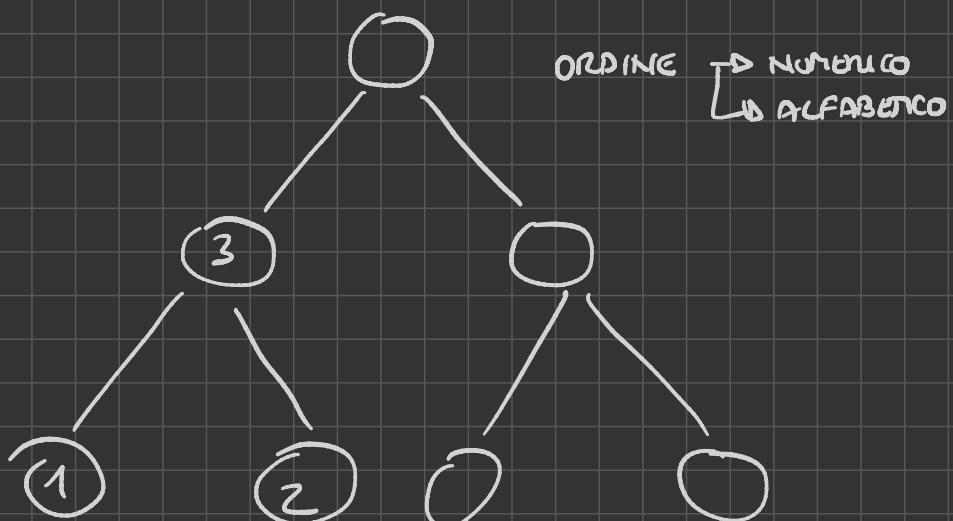
perché il foreach funziona deve mettere a disposizione un metodo che crea un iteratore ovvero , diventando ITERABLE

SET → INSIEMI  
    ↳ NO DUPLICATI  
    ↳ NON SONO INDICIZZATI → USARE ITERATOR

HASHSET → IMPLEMENTA HASH CODE

09/11/2023

CONCETTO DI ORDINE → ALCUNI BIANCIATI



NEI SET BISOGNA DEFINIRE UN ORDINAMENTO, QUELLO CHE PER NOI È IMPONIBILE  
PER IL NOSTRO DOMINIO

- ↳ METODO ITERATOR
- ↳ FANCI CHE I NOSTRI OGGETTI SIANO COMPARABILI

PUBLIC INT COMPARE TO (Object O)

STRATEGY → Parametro metodo → Costutture

TIPPI DI COLLECTION → MAPPE

15/11/2023

RENDERE LISTE IMMUTABILI CON IL SDK

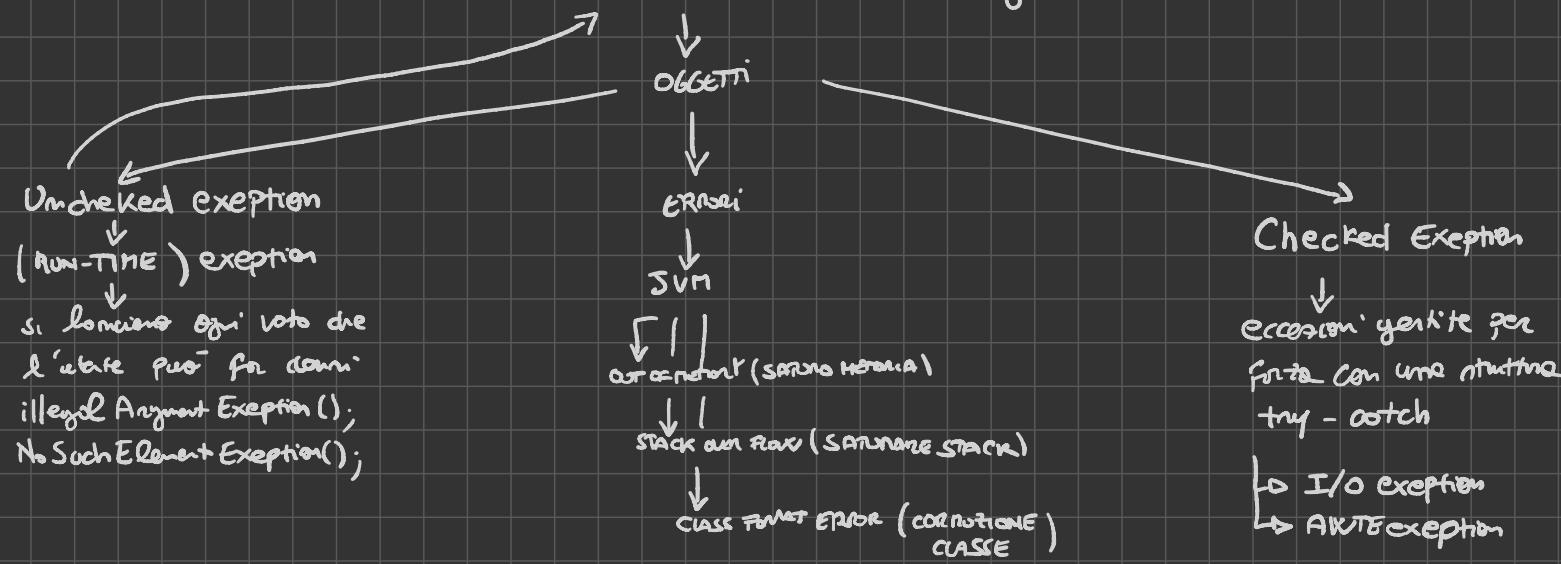
16/11/2023

VIOLAZIONI DI CONTRATTO D'USO OGGETTO  
↳ ERRORE GESTIBILI → CATTIVO UTILIZZO DEI METODI  
↳ ERRORE NON GESTIBILI

CAPIRE COME  
INTERCETTARE  
POSSIBILI ERRORE

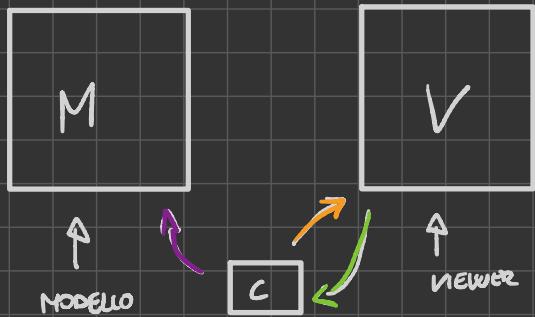
SEGNALARE  
SWEETI TIPI  
DI ERRORE  
E NON FAR  
PIANTARE IL  
PROGRAMMA

CONCLUSIVE ECCEZIONI → THROWN new Exception → JAVA. lang



22/11/2023

23/11/2023



Come interagiscono le due entità?

Si avrà un CONTROLLER

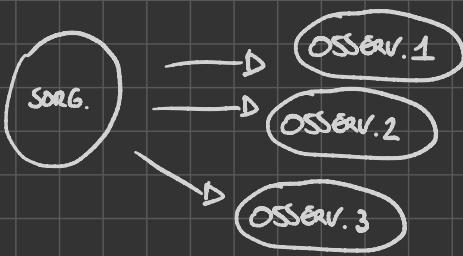
PARTE IL PROGRAMMA  
DALLA VIEW MANDA UN SEGNALE DI INIZIO GIOCO  
INIZIO DI GIOCO

Il modello e la view non devono interagire tra di loro

PROGETTARE LE 3 INTERFAZIE → DIVIDERE LE 3 INTERFAZIE IN PACKAGE → PACKAGE CON LETTERA MINUSCOLA

- METODO VOID
- METODO VOID
- METODO VOID

MODELLO METODO RESTITUISCONO METODI "DOMINIO" CHIAMATI DAI CONTROLLER

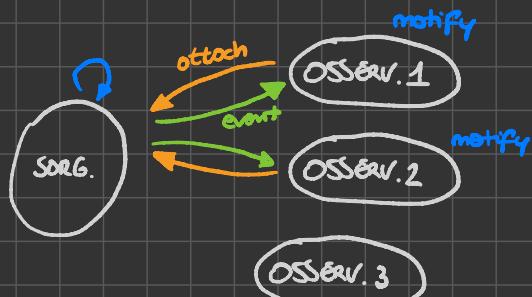


LA SORGENTE NOTIFICA DELLE COSE ALL'  
OSSERVATORI  
OSSERVATORI DEBONO REGISTRARSI ALL'EVENTO  
DELLA SORGENTE

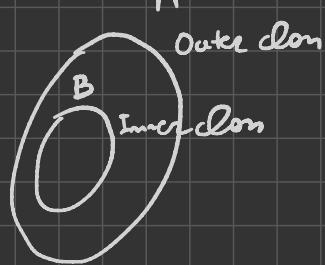
FASE 1 REGISTRAZIONE

FASE 2 EVENTO

FASE 3 NOTIFICA



24/11/2023



Inner class

private → B non può visibile all'  
esterno di A

✓ class A {

    class B {

} }

30/11/2023

MECCANISMI QUANTITATIVI LAMBDA E STREAM → PROGRAMMA FUNZIONALE

INTRODOTTO DA JAVA 8 → LAMBDA EXPRESSION → **LAMBDA CALCOLO**

$h(x) \rightarrow \lambda$  bonito nelle funzioni  
→ IMPATTA SULLE LIBRERIE

LAMBDA SI VASCIA IL TIPO DI DATO ALL'INGRESSO → RESTITUISCI IL VALORE RISULTANTE

$(T_1 \times_1, \dots, T_m \times_m) \rightarrow \{ \text{<body>} \}$

1/12/2023

INTERFACCIA SINGOLO METODO → INTERFACCIE FUNZIONALI

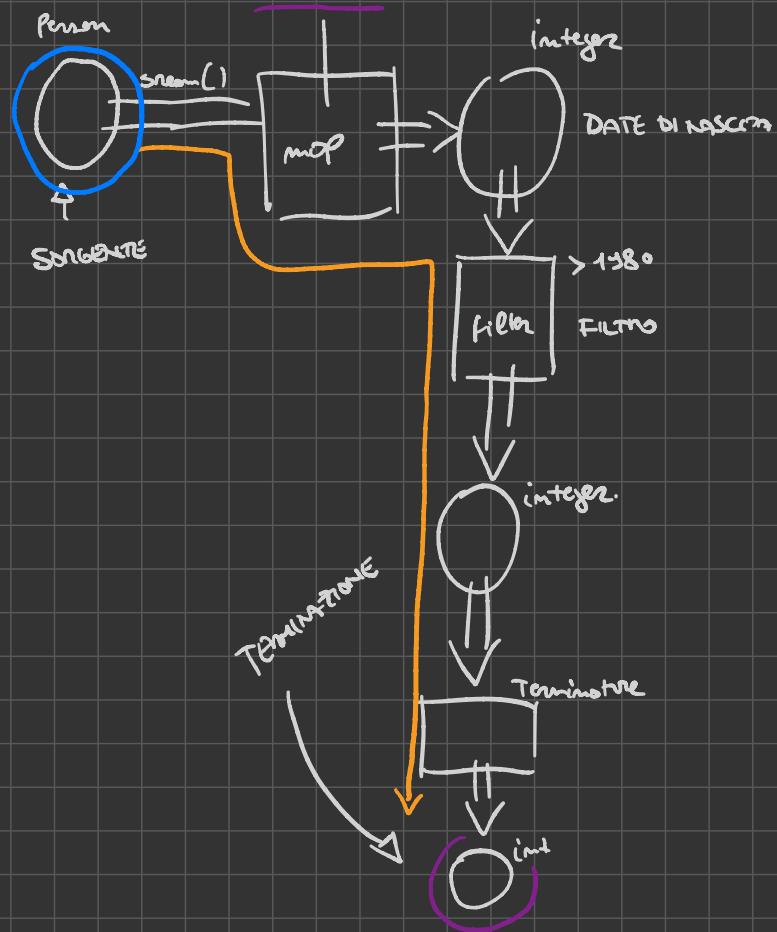
→ INTERFACCIA → PUNTA DICHIARAZIONE  
DI FUNZIONALITÀ  
→ JAVA MOLTE A DISPOSIZIONE ANCHE KEYWORD  
→ **DEFAULT**  
→ **STATIC** → PROPRÉTÉ  
DI MÉTODO  
↓  
CHE RENDE INIMPLEMENTABILI  
I METODI



7/12/2023

STREAM → CANALI CHE SPANNO  $1, \dots, N, \dots \infty$  DATI

STRUCTURE → SORGENTE  
→ SEQUENZA DI INFORMAZIONI  
→ TERMINAZIONE



12 | 12 | 2023

ANALISI → UML INTERFAZIE



DESIGN

