

## PROGRAMAZIONE ORIENTATA AGLI OGGETTI - MODULO 3

### Sommario

Cosa apprenderemo in questo modulo? .....	2
Modulo 3 – Ereditarietà.....	2
Ricapitolando !.....	3

### Cosa apprenderemo in questo modulo?

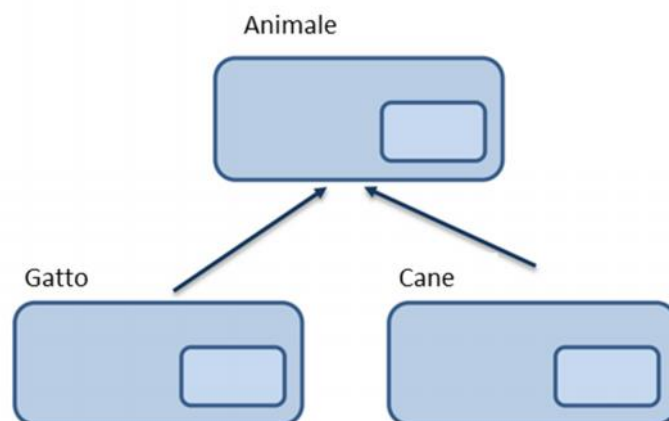
- Cos'è l'ereditarietà?
- A che serve?

### Modulo 3 – Ereditarietà

L'ereditarietà è un concetto molto importante nella programmazione OO e permette una migliore organizzazione e favorisce il riutilizzo del codice. Attraverso questo concetto le classi figlie condividono attributi e metodi con la propria classe madre.

Nel seguito, i termini "classe padre", "classe madre", "supertipo", "superclasse" e "classe base" saranno trattati come sinonimi ed indicheranno la classe originale da cui si eredita. I concetti di "classe figlia", "sottotipo", "sottoclasse" e "classe derivata" saranno trattati come sinonimi ed indicheranno le classi che ereditano il comportamento (attributi e metodi) di una classe madre.

Per rappresentare una classe che eredita il comportamento di un'altra classe, usiamo la parola riservata **"estende"**. Nell'esempio successivo "Gatto estende Animale", la classe Gatto è la classe figlia mentre la classe Animale è la classe madre. In questo modo, Gatto avrà tutti gli attributi pubblici della classe Animale e potrà anche utilizzare i suoi metodi pubblici (o anche ridefinirli, se necessario, come vedremo nel modulo del polimorfismo). Proprio per questo motivo diciamo che le classi figlie ereditano il comportamento della superclasse ma possono anche aggiungere altre caratteristiche (attributi) o nuove funzioni (nuovi metodi). È importante notare che l'ereditarietà segue anche i principi di visibilità, pertanto ciò che è privato in una classe non sarà visibile nelle sue sottoclassi.



Il concetto di ereditarietà è chiaramente legato al concetto di **"è un(a)"**. Nel nostro esempio precedente, gatto è un animale. L'ereditarietà riguarda anche il concetto di **generalizzazione** (ovvero si parte da una classe figlia e si risale fino alla classe madre) e **specializzazione** (ovvero si parte da una classe madre e si scende nella gerarchia fino alle classi figlie). Così la classe Gatto è una specializzazione della classe Animale e

## PROGRAMMAZIONE ORIENTATA AGLI OGGETTI - MODULO 3

viceversa la classe Animale è una generalizzazione della classe Gatto, come allo stesso modo potremmo dire del cane se definiamo che "Cane estende Animale".

Questo esempio illustra un altro punto importante: una superclasse può essere estesa in infiniti sottotipi, tuttavia, un sottotipo NORMALMENTE eredita le caratteristiche da una singola superclasse alla volta: quando un linguaggio permette che una classe erediti il comportamento da più di una superclasse, diciamo che tale linguaggio supporta "**l'ereditarietà multipla**".

Nell'esempio di cui sopra, le relazioni in pseudocodice delle classi sarebbero:

```
classe Animale {  
    // Qui sono definiti tutti gli attributi di un animale  
    // Qui sono definiti tutti i metodi di un animale  
}  
  
classe Gatto estende Animale {  
    // Qui possiamo dichiarare gli attributi specifici di un gatto  
    // Qui possiamo qui possiamo sovrascrivere i metodi di Animale, oppure  
    // dichiarare metodi specifici di un gatto  
}
```

### Ricapitolando !

In questa sezione abbiamo appreso **l'ereditarietà**, un altro pilastro della programmazione orientata agli oggetti. Vedere il modulo di seguito per ulteriori esempi sui principali punti di questo modulo.