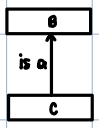
Ereditarietà in Java



la Sottoclasse eredita tutti i campi della superclasse, con la possibilità di ridefinire il comportamento di alcuni metodi ereditati e aggiungere altri metodi/campi propri

sintassi:

```
public class C extends B { ... }
```

In Java esiste un'unica modalità di derivazione: ogni metodo/campo viene derivato mantenendo il livello di visibilità della classe base

```
public class Rettangolo {
    protected int base;
    protected int altezza;
    public Rettangolo (int base, int altezza){
        this.base = base;
        this.altezza= altezza; }
    public final int area ()          //con questa keyword non posso
                                        usare il metodo nelle sottoclassi
        return base * altezza;}
    public int perimetro () {
        return (base + 2) + (altezza + 2);}
```

```
public class Quadrato extends Rettangolo {
    public Quadrato (int l) {
        super (l, l); }      //chiama il costruttore della superclasse
```

Recap:

	Classi in cui è dichiarato	Sottoclassi	All'esterno della classe	Nello stesso package
Public	✓	✓	✓	✓
Private	✓	✗	✗	✗
Protected	✓	✓	✗	✓
Package	✓	✗	✗	✓

Package non si riferisce a un singolo field/method

Tutte le classi hanno una superclasse Object e contiene alcuni metodi di utilità generale:

protected Object clone () ~> ritorna una shallow copy di this

public String toString() ~> ritorna una stringa che rappresenta la reference di this

public boolean equals (Object obj) ~> confronta la reference di this con la reference di obj

Classi di origine e di appartenenza

- Classe di origine: la classe usata nell'operatore new (è unica)
- Classe di appartenenza: classe d'origine e superclassi

Operatore instranceof

- Dato un oggetto o, è possibile controllare se C è una sua classe d'appartenenza

o instranceof C → Classe
↓
Oggetto

```
Quadrato q = new Quadrato (5);

q instranceof Rettangolo  // true

q instranceof Object      // true

q instranceof String      // false
```

```
public boolean equals (Object other) {
    if (other instranceof Rettangolo) {
    Rettangolo r = (Rettangolo) other;   // lo devo castare a rettangolo perché il confronto deve avvenire tra Rettangolo - Rettangolo e non Object - Rettangolo
    return r.l == l && r.h == h; }
    return false;
```