~ by Katsu Curry 

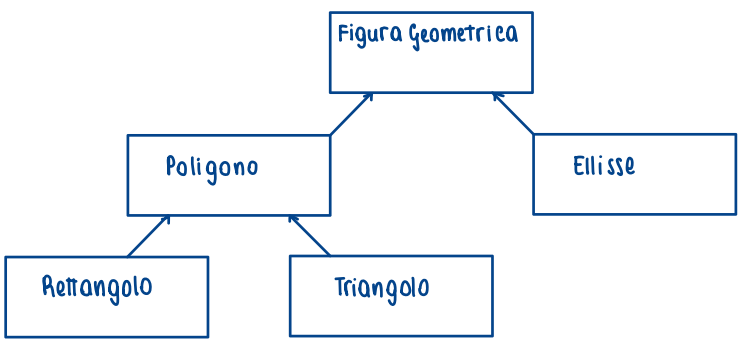
Polimorfismo

Definizione: il polimorfismo è la proprietà del codice di comportarsi in modo diverso in base al contesto di esecuzione.
In un linguaggio Object Oriented, il polimorfismo tra reference utilizza reference per riferirsi a oggetti di tipo dinamico diverso da quello dichiarato nel codice (tipo statico, compile-time).

OSS il tipo dinamico è limitato dalla corrispondenza tra tipi definita dalle gerarchie di ereditarietà. I tipi dinamici per un reference sono SOLO i sotto-tipi del tipo statico.

VANTAGGIO il codice scritto per un elemento della gerarchia può essere riutilizzato per tutti i suoi sotto-tipi.

ES: compatibilità tra tipi



Tipo statico	Tipo dinamico	
Poligono	p = new Rettangolo	✓
Poligono	p = new Figura Geometrica	✗
Figura Geom	F = new Triangolo	✓
Poligono	F = new Ellisse	✗

Binding Dinamico

Definizione il binding dinamico o late binding è il legame tra definizione e invocazione di un metodo non definito staticamente (compile-time) ma dinamicamente (run-time).

OSS il compilatore conosce il tipo dinamico di una variabile solamente a run-time, generando il codice per riconoscere il tipo effettivo (dinamico) di un oggetto e eseguire il metodo relativo.

ATTENZIONE
I seguenti non possono avere binding dinamico:
- metodi static → metodi associati a classi non oggetti, non è polimorfismo
- metodi final → non possono essere ridefiniti nelle sottoclassi
- overloading → puoi overridare i metodi, ma se ne esiste uno con parametro dello stesso tipo, invoca quello.

ES:
```
class Top {
    public String f(Object o) { return "Top"; }
}

class Sub extends Top {
    public String f(String s) { return "Sub"; }
    public String f(Object o) { return "SubObj"; }
}
```

```
public class Esempio {
    public static void main(String[] args) {
        Sub sub = new Sub();
        Top top = sub;
        String str = "Something";
        Object obj = str;

        System.out.println(top.f(obj));
        System.out.println(top.f(str));
    }
}
```

Stampa : SubObj
SubObj

OSS Perché? Top ha un solo metodo ⟶ f(Object) (firma del metodo)

```
Sub sub = new Sub();
Top top = sub;        ⟶ Il tipo dinamico di top è sub
String str = "Something";
Object obj = str;
```

Viene scelta l'implementazione di Sub ⟶ stampa SubObj

casting

Perché il polimorfismo sia possibile dobbiamo poter assegnare ad una variabile un dato di tipo diverso da quello dichiarato.

Definizione
- casting ⟶ coercizione di tipo
- up-casting ⟶ polimorfismo type-safe
- down-casting ⟶ polimorfismo type-unsafe

upcasting

Definizione coercizione di un tipo specializzato in un tipo più generico

OSS
- up-casting è sempre corretto (type-safe) e garantito dal compilatore
- dopo l'up-casting si possono invocare solo i metodi del tipo statico ma i metodi che vengono eseguiti sono del tipo dinamico (overriding)

ES
```
Studente s = new Studente()
Persona p = s
```
p ha solo i metodi in comune con s e seguono le firme dei metodi di s (in caso di override)

downcasting

Definizione | coercizione di un tipo generico in un tipo più specializzato (down, più in basso in gerarchia).

OSS il downcasting deve essere dichiarato esplicitamente dal programmatore.

ES (importanti)

1. Poligono p1 = new Rettangolo(); → solo metodi in comune, ma di tipo dinamico
   double b = ((Rettangolo) p1).getBase(); → per poter accedere ai metodi che sono SOLO della sottoclasse, serve il cast esplicito

2. Persona p = new Studente();
   Studente s = (Studente) p;

errori generati con Downcasting

runtime → java.lang.ClassCastException
↳ casting tra classi di gerarchie diverse, ma non posso riconoscerlo immediatamente.

compile time → java.lang.ClassCastException
↳ uso di tipi incompatibili riconosciuti immediatamente dal compilatore.

instanceof

Definizione l'operatore instanceof permette di controllare il tipo dinamico associato ad un referenee.

Sintassi | <reference> instanceof <Nome classe>
↳ true : tipo dinamico del referenze è dello stesso tipo o è un sottotipo.

Object

Tutte le classi in Java sono in relazione di ereditarietà con java.lang.Object

OSS | upcast a Object è sempre possibile
downcast a una qualsiasi classe è accettato dal compilatore, ma attenzione alle classi.

getClass()

Definizione | getClass() è un metodo final della classe Object e restituisce una rappresentazione del tipo dinamico dell'oggetto.