



Argomenti di oggi

Polymorphism
Type conversion (cast)
Upcast & Downcast



Polimorfismo

```
class OP {  
    int f(int a,int b){  
        return a*b;  
    }  
}  
Class Somma extends OP {  
    int f(int a,int b){  
        return a+b;  
    }  
}  
class Sottrazione extends OP {  
    int f(int a,int b){  
        return a-b;  
    }  
}
```



Polimorfismo

```

class Test {
    public static void main(String[] a) {
        new Test;
    }
    public Test() {
        OP o;
        ...
        if (i!=0) o=new Somma();
        else o=new Sottrazione();
        ...
        o.f(3,2);
    }
}
  
```

Che metodo viene chiamato qui?



Static and Dynamic binding

Cosa succede quando si chiama un metodo su di un oggetto
(esempio: C obj; ... obj.f(args)) ?

- 1.il compilatore cerca tra i metodi dell'oggetto obj i metodi f(...) e li enumera
- 2.il compilatore determina i tipi di parametri passati. Se trova un match unico con i metodi della classe C (anche tramite cast!) ne prende nota – **overloading resolution** – altrimenti genera un messaggio di errore
- 3.se il metodo è private, static, final o un costruttore il compilatore sa esattamente che metodo chiamare (**static binding**)
- 4.altrimenti il metodo dipende da qual'è la classe a cui obj appartiene a runtime ed il compilatore deve delegare alla JVM la determinazione del metodo a run time (**dynamic binding**)



binding

```
Class A {...}  
Class B extends A {...}  
A anObject=new A();  
A anObject=new B()
```

REGOLA

La referenza di una classe può contenere oggetti delle sottoclassi.

... perché l'istanza della sottoclasse risponde a tutti i metodi della superclasse.



Static and Dynamic binding

Se il programma usa il dynamic binding, la JVM deve chiamare il metodo appropriato al particolare tipo di oggetto obj.

- Es. **obj** sia di tipo D, classe derivata da C. Se esiste un metodo **f(int)** in D, questo sarà il metodo chiamato, altrimenti verrà chiamato il metodo **f(int)** di C
- Eseguire ogni volta questi controlli non è efficiente, per cui la virtual machine calcola in anticipo un method table per ogni classe che raccoglie tutte le signatures dei metodi



Fondamenti di Java

Una Pila Polimorfa



La Pila in Java - 1

```
package strutture;  
public class Pila {  
    int size;  
    int defaultGrowthSize;  
    int marker;  
    Object contenuto[];  
    final int initialSize=3;  
  
    Pila() {  
        size=initialSize;  
        defaultGrowthSize=initialSize;  
        marker=0;  
        contenuto=new Object[size];  
    }  
}
```



La Pila in Java - 2

Abilita lo
static
binding

Fac.Scienze - Università di Trento

```
final void inserisci(Object k) {  
    if (marker==size)  
        {cresci(defaultGrowthSize);}  
    contenuto[marker]=k;  
    marker++;  
}  
  
Object estrai() {  
    assert (marker>0):"Estrazione da Pila vuota";  
    return contenuto[--marker];  
}
```



La Pila in Java - 3

Abilita lo
static
binding

Fac.Scienze - Università di Trento

```
private void cresci(int dim){  
    Object temp[ ]=new Object[size];  
    for (int k=0;k<size;k++)  
        temp[k]=contenuto[k];  
    contenuto=new Object[size+defaultGrowthSize];  
    for (int k=0;k<size;k++)  
        contenuto[k]=temp[k];  
    size+=defaultGrowthSize;  
}
```



Classi Wrapper

Adesso che la Pila contiene oggetti, come faccio ad usarla per degli int?

Soluzione:
uso una
classe "wrapper"



int	Integer
float	Float
char	Char
byte	Byte
double	Double

Generano oggetti che hanno come variabile di istanza un tipo di dato primitivo con nome uguale

Integer

int

Float

float



La Pila in Java - 5

```
public static void main(String args[]) {
    int dim=10;
    Pila s=new Pila();
    for (int k=0;k<dim;k++){
        Integer o=new Integer(k);
        s.inserisci(o);
    }
    for (int k=0;k<3*dim;k++) {
        Integer i = s.estrai();
        int w=i.intValue();
        System.out.println(w);
    }
}
```



La Pila in Java - 5

```
public static void main(String args[]) {  
    int dim=10;  
    Pila s=new Pila();  
    for (int k=0;k<dim;k++){  
        Integer o=new Integer(k);  
        s.inserisci(o);  
    }  
    for (int k=0;k<3*dim;k++) {  
        Integer i = s.estrai();  
        int w=i.intValue();  
        System.out.println(w);  
    }  
}
```

ERRORE!
Non posso
mettere un
Object in un
Integer!



La Pila in Java - 6

```
public static void main(String args[]) {  
    int dim=10;  
    Pila s=new Pila();  
    for (int k=0;k<dim;k++){  
        Integer o=new Integer(k);  
        s.inserisci(o);  
    }  
    for (int k=0;k<3*dim;k++) {  
        Integer i = (Integer)s.estrai();  
        int w=i.intValue();  
        System.out.println(w);  
    }  
}
```



La Pila in Java - 7a

```
public static void main(String args[]) {  
    int dim=10;  
    Pila s=new Pila();  
    //INSERIMENTO  
    for (int k=0;k<dim;k++){  
        Object o;  
        if (Math.random()<0.5)  
            o=new Integer(k);  
        else  
            o=new Float(k*Math.PI);  
        s.inserisci(o);  
    }  
}
```



La Pila in Java - 7b

```
// ESTRAZIONE  
for (int k=0;k<dim;k++) {  
    Object o = s.estrail();  
    if (o instanceof Integer) {  
        Integer i = (Integer) o;  
        int w = i.intValue();  
        System.out.println("an int:"+w);  
    } else if (o instanceof Float) {  
        Float i = (Float) o;  
        float w = i.floatValue();  
        System.out.println("a float:"+w);  
    } else  
        System.out.println("Unknown class!");  
}  
}
```




La Pila in Java - 7c

OUTPUT:

```
a float:28.274334
an int:8
an int:7
a float:18.849556
an int:5
an int:4
a float:9.424778
a float:6.2831855
a float:3.1415927
a float:0.0
```