Orientação a Objetos

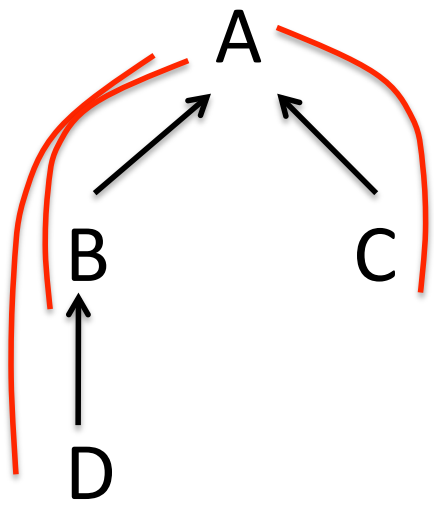
Prof. Guido Araujo
www.ic.unicamp.br/~guido

Object Layout

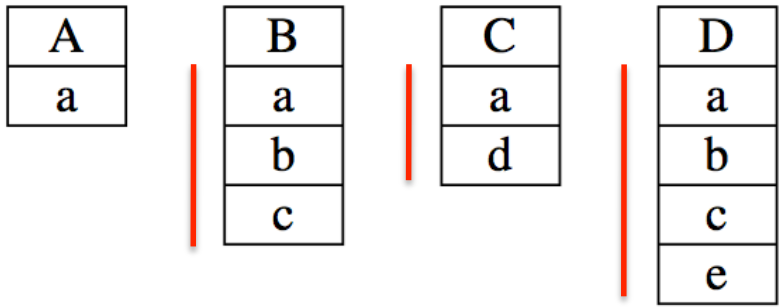
```
class A {
   int x;
   float y;
   void  f();
}

main() {
     :
  A a = new A()
     :
   a.x = 0;
   a.y = 3.1;
   a.f(y);
}
```

Herança Simples



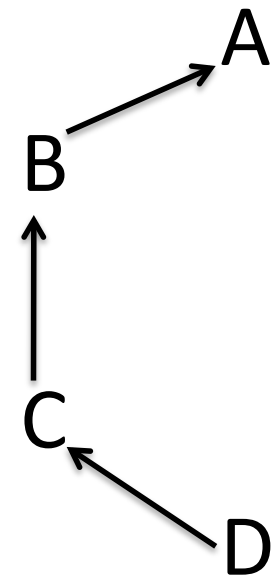

```
class A {            int a = 0;}
class B extends A {int b = 0;
                     int c = 0;}
class C extends A {int d = 0;}
class D extends B {int e = 0;}
```



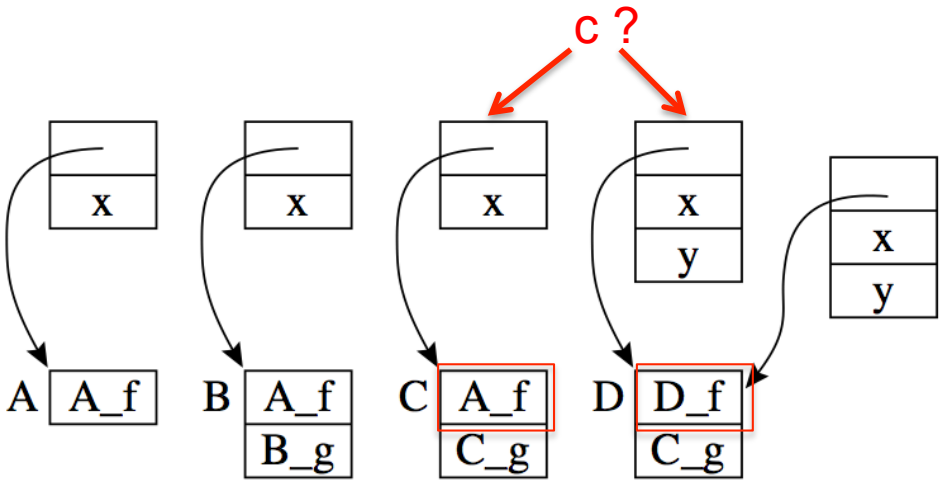
Métodos Dinâmicos com Sobrecarga

```
main() {
   :
  if (k < i)
     c = d;
  c.f();
}
```

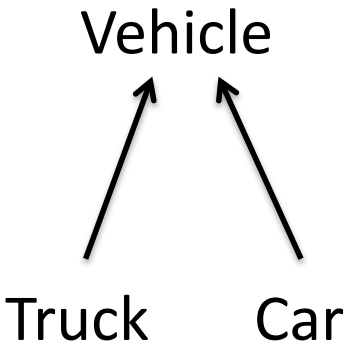
Qual o f()?




```
class A                {int x = 0;
                        int f() {...} }
class B extends A      {int g() {...} }
class C extends B      {int g() {...} }
class D extends C      {int y = 0;
                        int f() {...} }
```



Exemplo Java




```
class Vehicle {
    int position;
    void move (int x) { position = position + x; }
}
class Car extends Vehicle{
    int passengers;
    void await(Vehicle v) {
        if (v.position < position)
            v.move(position - v.position);
        else
            this.move(10);
    }
}
```

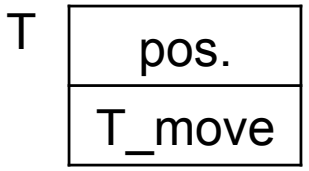
V

pos.
V_move

C

pos.
V_move
pass.
C_await

Exemplo Java



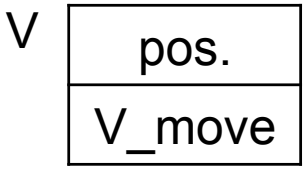

```
class Truck extends Vehicle{
    void move(int x) {
        if (x <= 55) { position = position + x; }
    }
}
class Main{
    public static void main(String args[]) {
        Truck t = new Truck();
        Car c = new Car();
        Vehicle v = c;
        c.passengers = 2;
        c.move(60);
        v.move(70);
        c.await(t);
    }
}
```

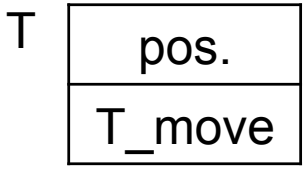
Class Descriptor com Herança

```
class Main{
    public static void main(String args[]) {
        Truck t = new Truck();
        Car c = new Car();
        Vehicle v = c;
        c.passengers = 2;
        c.move(60);
        v.move(70);
        c.await(t);
    }
}

class Car extends Vehicle{
    int passengers;
    void await(Vehicle v) {
        if (v.position < position)
            v.move(position - v.position);
        else
            this.move(10);
    }
}
```

Qual o move()?



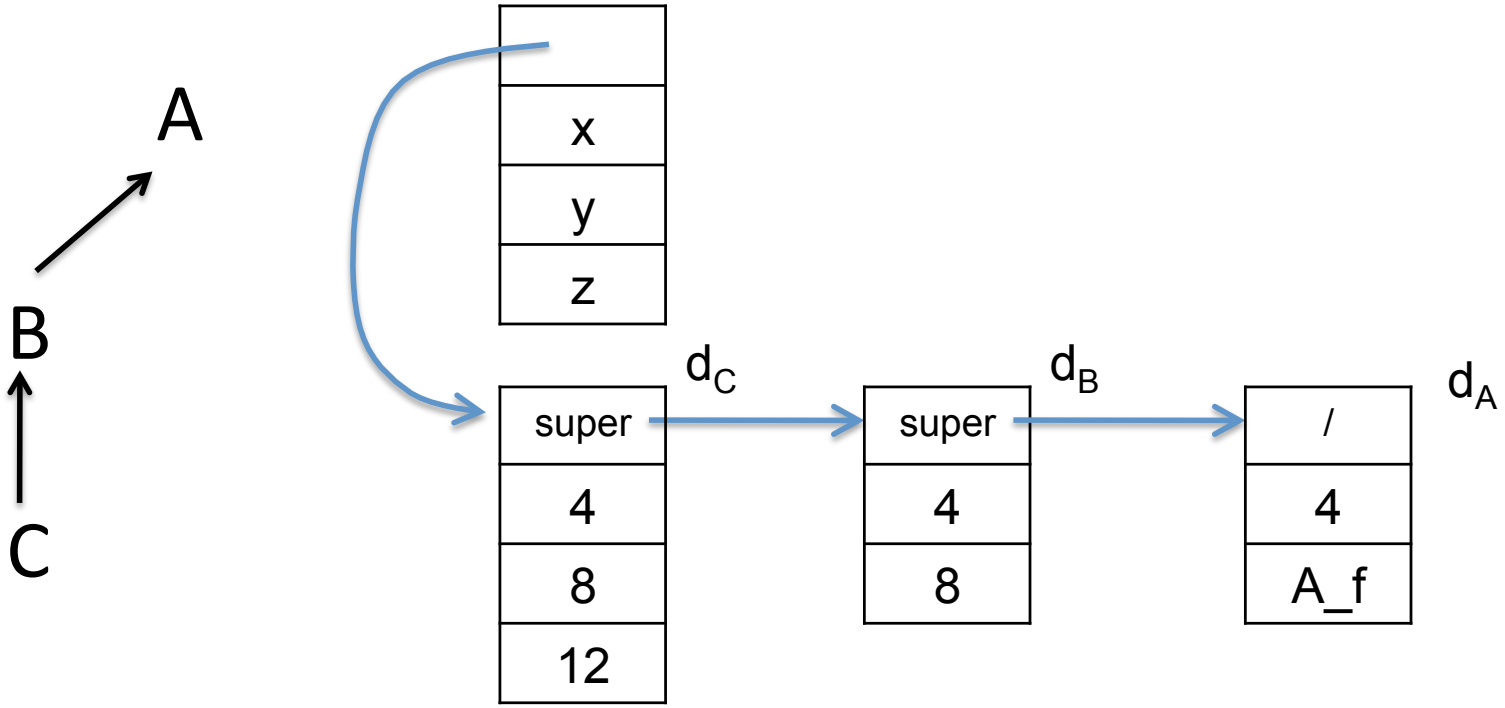


Métodos Estáticos

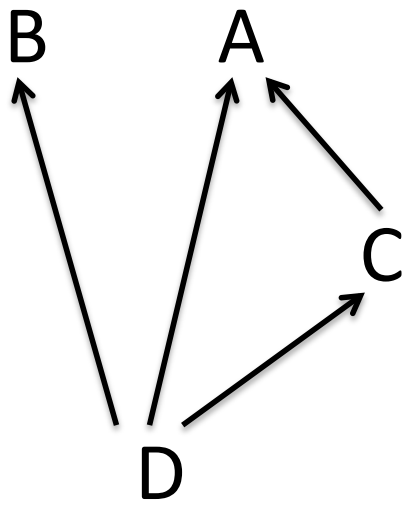
```
class A {
   int x;
   static void  f()
}
```

```
class B {
   int y;
}
```

```
class C {
   int z;
}
```



Herança Múltipla



Como alocar os campos?

A
a

B
b
c

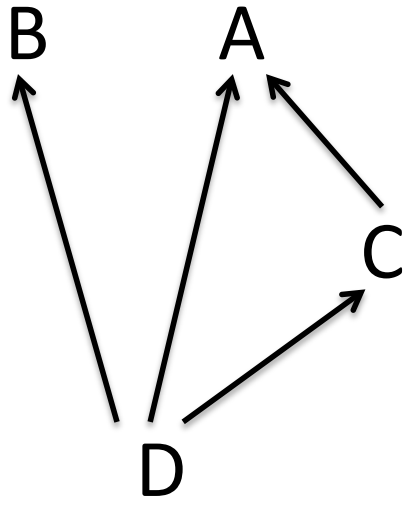
C
a
d

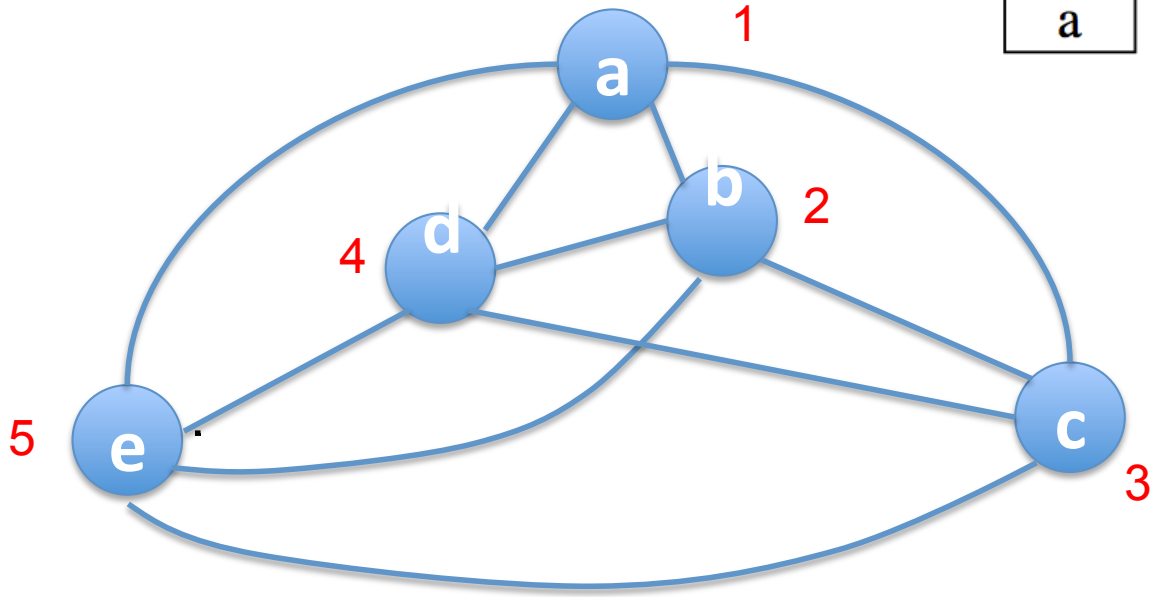
D
a
b
c
d
e

```
class A { int a = 0; }
class B { int b = 0;
          int c = 0; }
class C extends A { int d = 0; }
class D extends A,B,C { int e = 0; }
```

Usando Coloração

```
class A { int a = 0; }
class B { int b = 0;
          int c = 0; }
class C extends A { int d = 0; }
class D extends A,B,C { int e = 0; }
```





A
a

B
b
c

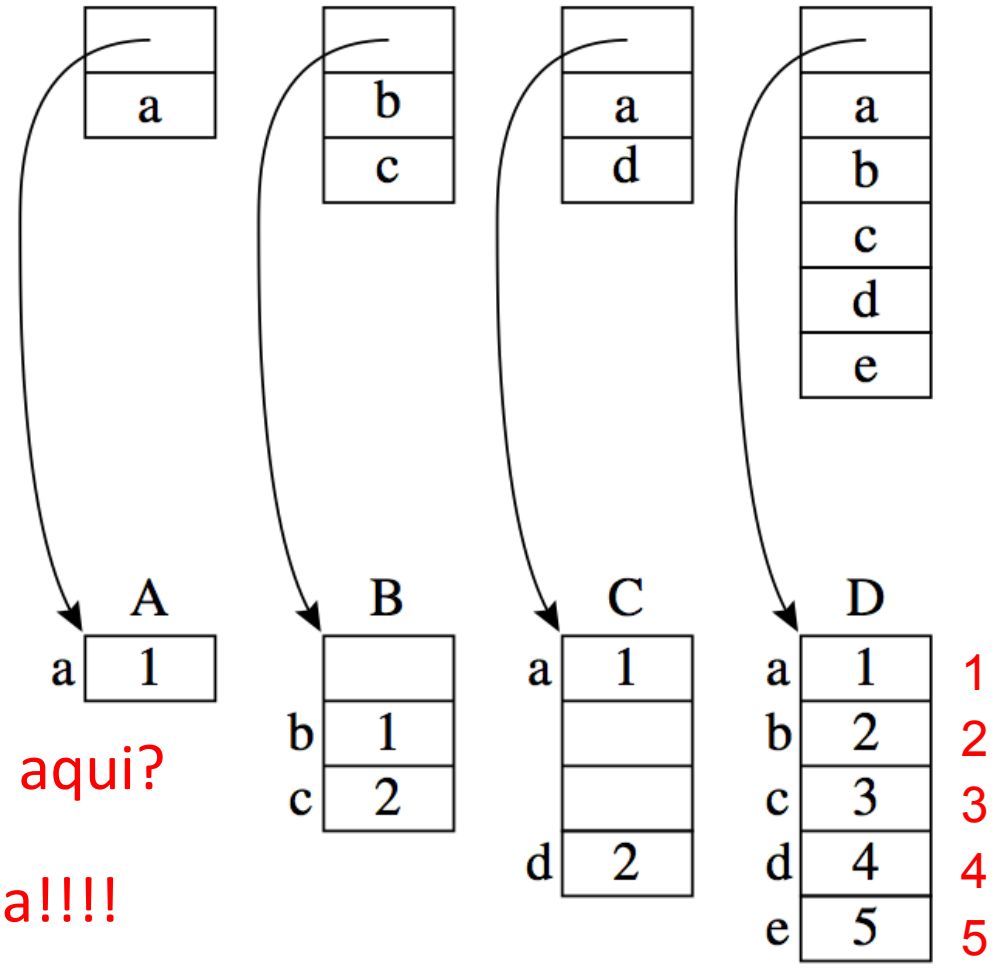
C
a
d

D	
a	1
b	2
c	3
d	4
e	5

Qual o problema aqui?

Objetos têm buracos!!

Eliminando Buracos



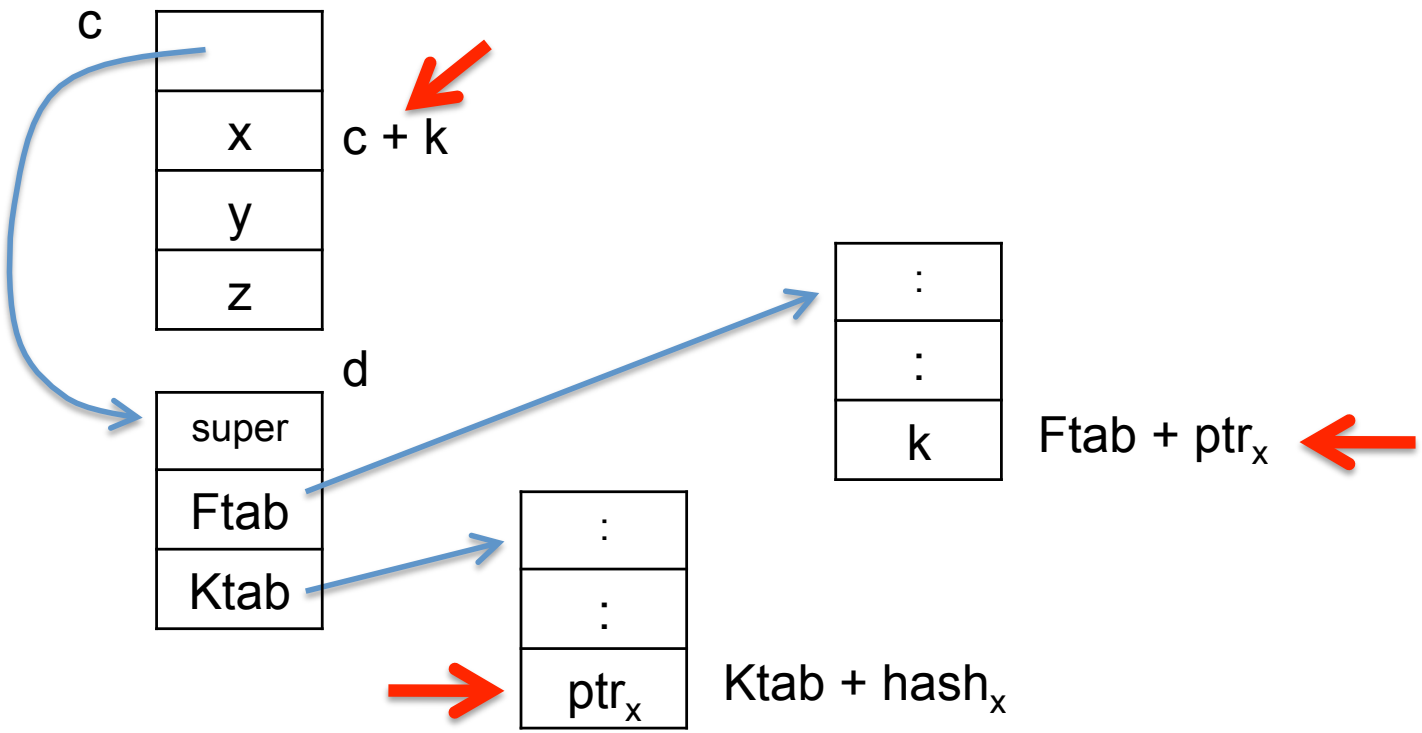
Qual o problema aqui?

Herança dinâmica!!!!

Requer coloração em tempo de link

Evitando Coloração

1. Fetch the class descriptor d at offset 0 from object `c`.
2. Fetch the field name f from the address offset $d + \texttt{Ktab} + \text{hash}_x$.
3. Test whether $f = \text{ptr}_x$; if so
4. Fetch the field offset k from $d + \texttt{Ftab} + \text{hash}_x$.
5. Fetch the contents of the field from $c + k$.



Teste de Classe

```
if (k < i)
    b = c;
```

Java:

```
if (b instanceof C)
     f((C)b)
else ...
```

$$
\begin{aligned}
& t_1 \leftarrow x.\text{descriptor} \\
L_1: \quad & \textbf{if } t_1 = C \textbf{ goto } \textit{true} \\
& t_1 \leftarrow t_1.\text{super} \\
& \textbf{if } t_1 = \text{nil } \textbf{goto } \textit{false} \\
& \textbf{goto } L_1
\end{aligned}
$$