Università degli Studi della Campania *Luigi Vanvitelli*

Università di degli Studi della Campania Luigi Vanvitelli Dipartimento di Ingegneria

Programmazione ad Oggetti

a.a. 2020-2021

Upcasting

Docente: Prof. Massimo Ficco

E-mail: *massimo.ficco@unicampania.it*

Upcasting

La funzionalità più importante dell'ereditarietà non è la possibilità di poter aggiungere funzionalità ad una classe

La cosa più importante è che **un oggetto della *classe derivata è anche un oggetto della classe base***

Tale definizione è supportata dal linguaggio nella pratica!!!!!

Upcasting

```
// Un messaggio inviabile alla classe base può essere inviato anche alla classe derivata
class Instrument {
    public void play() {System.out.println("Play Instrument");}
    static void tune(Instrument i) {i.play();}
}

// Un oggetto Wind eredita l'interfaccia della classe base
public class Wind extends Instrument {
       public  void play(){System.out.println("Play Wind");
       static void tune(Wind i) {i.play();}

   public static void main(String[] args) {
       Wind flute1 = new Wind();
       flute1.play(); → "Play Wind"
       flute1.tune(flute1); → "Play Wind"

       Instrument flute2 = new Wind(); // Upcasting
       flute2.play();   // Upcasting → "Play Wind"
       Wind.tune(flute2); // Upcasting → "Play Wind"
       Instrument.tune(flute2); // Upcasting → "Play Wind"

Instrument pfd = new Instrument();
       Instrument.tune(pfd); // → "Play Instrument
}}
```

*Altro esempio

La classe **Vector** del package java.util può contenere un oggetto di qualunque classe

Perché ?

…Tutte le classi Java ereditano implicitamente la classe Object

Andare a controllare nelle Java API qual è l'interfaccia della classe Object

V:

A cura del

Prof. Massimo Ficco

e del

Prof. Salvatore Venticinque