Java Method Overriding

- A method defined in a super class may be overridden by a method of the same name defined in a sub class
- The overriding method has the same name, number and type of parameters, and return type as the method that it overrides

Πλεονεκτήματα Κληρονομικότητας

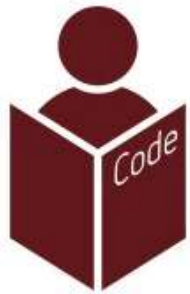
- Μικρότερο μέγεθος για τις τάξεις – παιδιά
- Εύκολη τροποποίηση των κοινών μεθόδων και ιδιοτήτων
- Επεκτασιμότητα
- Base class code testing
- Λογική «τμηματοποίηση» του κώδικα

Χρήση μεταβλητών αντικειμένων

Μεταβλητές

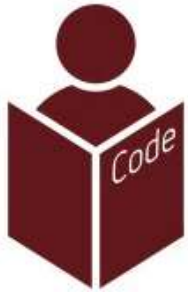
- Μπορούμε να αναφερθούμε σε μια μεταβλητή αντικειμένου στη Java με περισσότερους από έναν τρόπους:
  - Απευθείας με την τάξη στην οποία ανήκει
  - Μέσω μιας υπερ-τάξης της τάξης που ανήκει
  - Μέσω ενός interface το οποίο υλοποιεί
  - Μέσω της τάξης Object (Γιατί?)
- Σε κάθε περίπτωση, ωστόσο, πρέπει να προσέξουμε τις διαφορές που προκύπτουν

Παράδειγμα



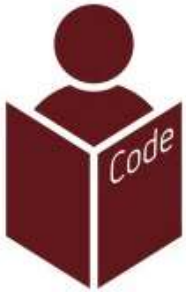

```
class Human{
    int age;
    String name;
}
interface ISpeak{
    void speak(String s);
}
class Student extends Human implements ISpeak{
    int am;
    @Override
    public void speak(String s) {
        System.out.println("Hello "+s);
    }
}
class Professor extends Human implements ISpeak{
    int officeNumber;
    @Override
    public void speak(String s) {
        System.out.println("Hi "+s);
    }
}
```

Μέσω της τάξης του




```
Student student = new Student();
student.name = "George";
student.am = 12345;
student.speak( s: "Unipi");
```

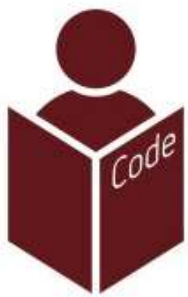
Μέσω της super class




```
    Human human = new Student();
    human.name = "Maria";
    human.am = 23456;
    human.speak("Informatics");
}
```

Cannot resolve method 'speak(java.lang.String)'

Μέσω του interface που υλοποιεί



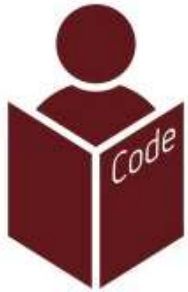

```
ISpeak speaker = new Student();
speaker.speak( s: "Piraeus");
speaker.name = "Peter";
speaker.am = 34567;
```

Cannot resolve symbol 'am'

Μέσω της τάξης Object


```
Object o = new Student();
o.name = "Costas";
o.am = 45678;
o.speak("Greece");
```

Γιατί υπάρχει τότε αυτή η δυνατότητα;




```
ISpeak[] speakers = new ISpeak[2];
speakers[0] = new Student();
speakers[1] = new Professor();
for (ISpeak iSpeak : speakers)
    iSpeak.speak( s: "University of Piraeus!");
```

Casting with objects

What is Casting

- Casting ονομάζουμε τη διαδικασία κατά την οποία «αναγκάζουμε» μια μεταβλητή να «συμπεριφερθεί» ως μια άλλη μεταβλητή
- Σε επίπεδο τάξεων το casting λαμβάνει χώρα όταν ένα αντικείμενο είναι ενός άλλου τύπου (IS-A), δηλαδή έχουμε σχέση κληρονομικότητας

Παράδειγμα

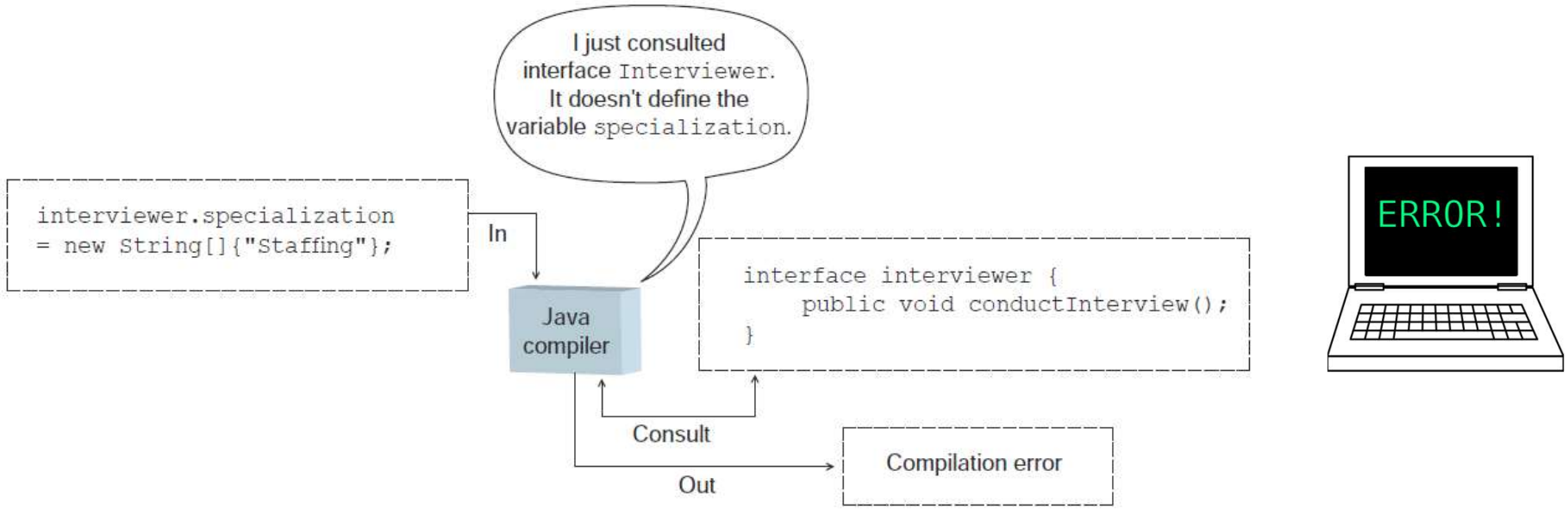
```
class Employee {}
interface Interviewer {
    public void conductInterview();
}
class HRExecutive extends Employee implements Interviewer {
    String[] specialization;
    public void conductInterview() {
        System.out.println("HRExecutive - conducting interview");
    }
}
class Manager implements Interviewer{
    int teamSize;
    public void conductInterview() {
        System.out.println("Manager - conducting interview");
    }
}
```

Δοκιμές 1/2

```
Interviewer interviewer = new HRExecutive();

interviewer.specialization = new String[] {"Staffing"};
```

⟵ **Won't compile**



Δοκιμές 2/2

```
((HRExecutive)interviewer).specialization = new String[] {"Staffing"};
```