dangling ELSE

Da in einer *if*-Anweisung das *else* fehlen darf, ist bei geschachtelten *if*-Anweisungen unklar, wohin ein *else* gehört:

if (door.isOpen())

if (resident.isVisible())

resident.greet("Hallo!");

else door.bell.ring();

JAVA legt fest, dass das *else* zur innerst möglichen *if*-Anweisung gehört!

Zugriffskontrolle

- Eine Variable kann nur verwendet werden, wenn ihr Typ (die Klasse, die ihren Wertebereich angibt) zugreifbar ist und sie selbst zugreifbar ist.
- Eine Methode kann nur verwendet werden, wenn sie selbst zugreifbar ist und die Klasse, für deren Objekte die Methode Handlungen bereitstellt.
- Ebenso kann ein Konstruktor nur verwendet werden, wenn er selbst und die Klasse, für die er Objekte erzeugt, zugreifbar ist.

Modifikatoren

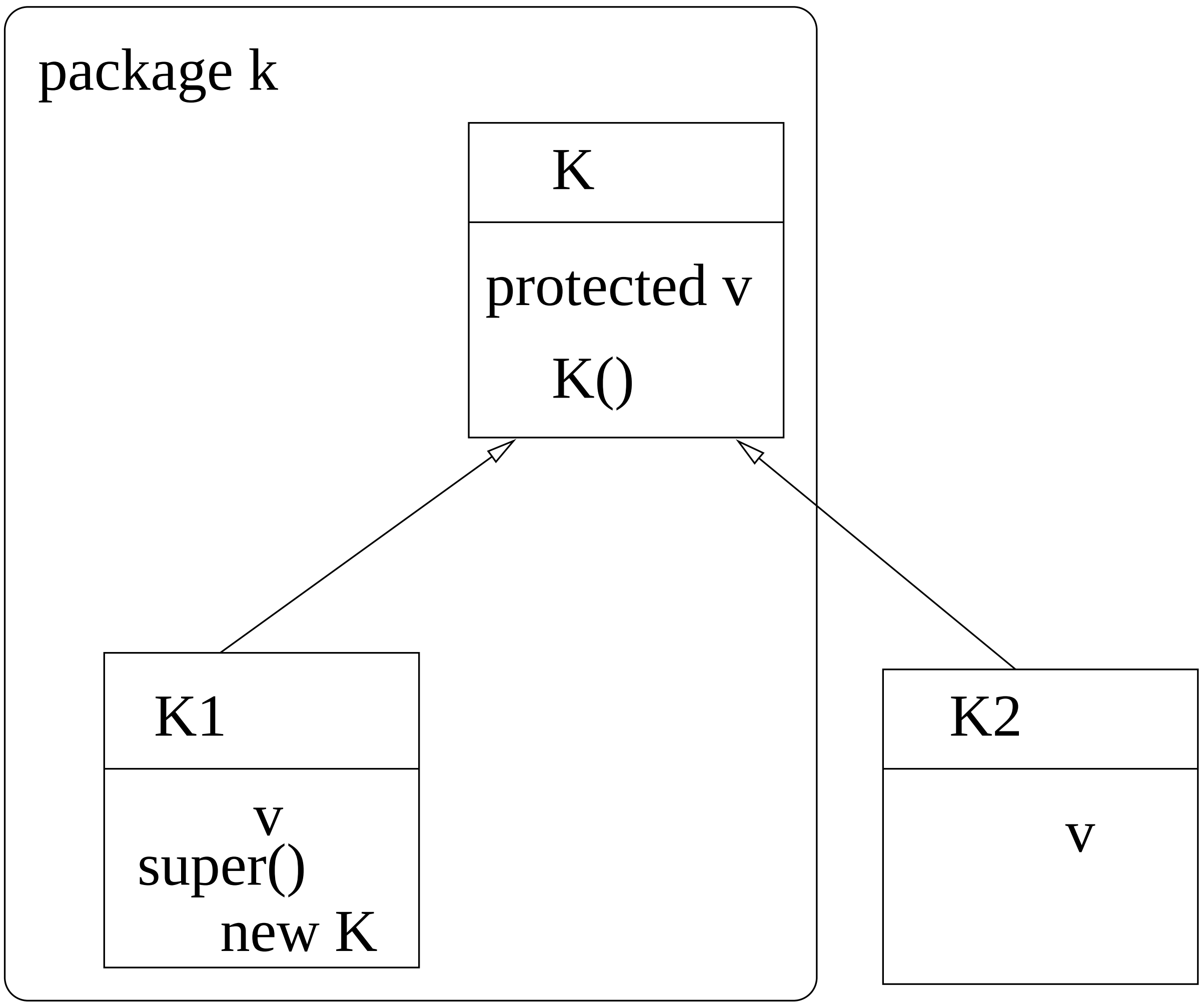
public : weltweit zugreifbar;

ohne Schlüsselwort : von dem Paket aus zugreifbar, in dem die Klassendeklaration steht;

protected : von dem Paket aus zugreifbar, in dem die Klassendeklaration steht, Variablen auch von Unterklassen außerhalb des Pakets aus;

private : nur innerhalb der Klasse zugreifbar, in der die Variable oder Methode deklariert ist. Also auch nicht erblich!

Beispiel



Sichtbarkeit von Variablen

Eine Variable kann

- eine Klasseneigenschaft – geschrieben mit dem Schlüsselwort $static$,
- eine Objekteigenschaft – deklariert am Anfang von $ClassBody$ (ohne $static$),
- ein Unikat – eine Variable von einem einfachen Datentyp,
- eine Hilfsgröße, die wir gerade mal (z.B. in einer Methode oder in einer Schleife) benötigen

ausdrücken.

Eine Klasseneigenschaft ist überall sichtbar, wo die Klasse sichtbar ist. Eine Objekteigenschaft ist ebenfalls überall sichtbar, wo die Klasse sichtbar ist, deren Objekte diese Eigenschaft haben. Ob die Klasse sichtbar (zugreifbar) ist, ergibt sich daraus, in welchem Paket und mit welchem (oder keinem) Modifikator sie deklariert wurde.

Lokale Variablen

gelten nur innerhalb des Blocks, in dem sie stehen.

Eine for-Anweisung wird wie ein Block behandelt.

Innerhalb dieses Geltungsbereichs darf der Name der lokalen Variablen nicht noch einmal auftreten. Beispielsweise darf in dem Block, in dem die lokale Variable deklariert ist, nicht noch eine for-Schleife mit einer Variable gleichen Namens vorkommen.

Außerhalb schon.

Nicht geht:

```
class SichtTest1 {
    static int x=7;
    public static void main (String[] args) {
        int x = x;
    }
}
```

SichtTest1.java:8: Variable x may not have been initialized. int x = x;

— 1 error

Compilation exited abnormally with code 1 at Thu Nov 12 10:40:50

Dies geht:

```
class SichtTest2 {
    static int x=7;
    public static void main (String[] args) {
        int x = (x=2)*2;
        System.out.println (x);
    }
}
```

Ergebnis: 4

Verbergen von Namen durch lokale Variablen

Ist ein als lokale Variable bereits als Variablenname deklariert, wird er im Gültigkeitsbereich der lokalen Variable überdeckt. Er kann aber noch mit *this* angesprochen werden.

```
class Pair {

    Object first, second;

    public Pair (Object first, Object second) {

        this .first = first;
        this .second= second;
    }

}
class Pairtest {
    public static void main (String argv[]) {
        Pair pair = new Pair ("Adam","Eva");
        System.out.println (pair.first + ""+ pair.second);
    }
}
```

Ergebnis: Adam Eva

```
import ballbeispiel.*; import AlgoTools.IO;
class Mensch { String name, geschlecht; Hausrat hausrat; } ;
class Studierend extends Mensch {
    int semester,monat,jahr;
    public Studierend () {
        super ();
        semester = IO.readInt ("Im wievielten Semester ist Stud? ");
        monat = IO.readInt ("Der wievielte Monat ist jetzt?");
        jahr = IO.readInt ("In welchem Jahr? ");
    }
    public void studieren () {
        for (int i=this.monat; 13 >i;i++) {      //Monate zaehlen
            if ((i!=this.monat) && (i==4 | i==10)) {
                this .semester++;              //Semester zaehlen
                System.out.println (this .name+"ist "+this.jahr
                +"im "+semester+". Semester");   }
        }
        this .jahr++;                                  //Jahre zaehlen
        this .monat=1;
        if (9>semester)           //Studienende noch nicht erreicht?
            studieren ();                      //dann weiterstudieren
        else System.out.println ("Das Diplom!");     //sonst Diplom
    }
}
class Studi {
    private static void main (String argv[]) {
        Studierend stud;
        stud = new Studierend ();
        stud.studieren ();
        System.out.println (stud.name+ "diplomiert "+stud.jahr);
    }
}
```

Probieren Sie einmal aus, was passiert, wenn Sie statt i den Variablennamen $monat$ verwenden!