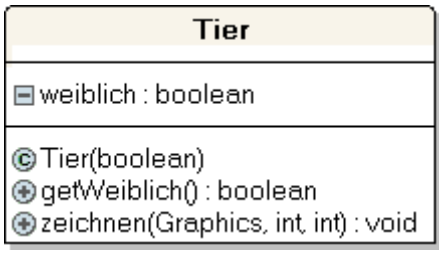
Vererbung – Leseübung: Lösung

AvHG, Inf, My

Es existiert eine Klasse Tier, die folgendermaßen aussieht:



Von der Klasse Tier wird eine Klasse Hund abgeleitet. Fülle die Lücken im Code geeignet aus:

```
import java.awt.Graphics;

public class Hund extends Tier {

    // Aufgabe (d) unten
    public Hund() {
        super(false);
    }

    // Aufgabe (c) unten
    public Hund(boolean weiblich) {
        super(weiblich);
    }

    public void zeichnen (Graphics g, int x, int y) {
        // Nur männliche Hunde sollen gezeichnet werden. Weibliche Hunde
        // verstecken sich. Rufe die Methode zeichnen aus der Superklasse
        // Tier auf, wenn der Hund männlich ist.
        if (getWeiblich() == false) {
            super.zeichnen(g, x, y);
        }
    }
}
```

Fragen:

(a) Wieso gibt es in der überschriebenen Methode `zeichnen(...)` eine Fehlermeldung, wenn man auf die Variable weiblich zugreift?

✔ Weil `weiblich` in der Superklasse als private deklariert ist. Es gibt jetzt zwei Lösungswege:

1. Man benutzt die öffentliche Methode `getWeiblich()`, oder
2. man ändert die Deklaration des Attributs weiblich in der Superklasse von `private` auf `protected` (oder verzichtet ganz auf einen Modifier – dann können innerhalb des Packages alle Klassen auf dieses Attribut zugreifen).

(b) Wieso gibt der Compiler eine Fehlermeldung aus, wenn in der Klasse Hund kein Konstruktor steht? Schreibe zur Beantwortung der Frage den Konstruktor auf, den das System automatisch erzeugt:

✔
```
public Hund() {
    super();
}
```

(c) Füge in die Klasse Hund einen Konstruktor ein, der als Parameter einen boolschen Wert erhält. Der boolsche Wert gibt an, ob der Hund weiblich ist oder nicht. Dies ist derselbe Konstruktor wie ihn die Superklasse Tier besitzt.

(d) Schreibe zusätzlich einen parameterlosen Konstruktor für die Klasse Hund, der immer männliche Hunde erzeugt.