Klassendefinitionen in Java

Der Kopf einer Klassendefinition

Der Kopf einer Klassendefinition besteht aus der *Sichtbarkeit*, dem Schlüsselwort `class` und dem Namen der Klasse, der immer mit einem Großbuchstaben beginnt. Das Innere der Klassendefinition ist in geschwungene Klammern eingeschlossen:

```
Sichtbarkeit  Name der Klasse
public  class Apfel
{
   <Datenfelder>
   <Konstruktoren>
   <Methoden>
}
```

Das Innere der Klassendefinition ist üblicherweise in drei "Abschnitte" gegliedert.

Datenfelder

Jedes Attribut einer Klasse wird in einer speziellen Variablen *(Datenfeld)* gespeichert. Die Definition eines Datenfeldes besteht aus der Sichtbarkeit, dem Datentyp und dem Namen:

```
Sichtbarkeit    Datentyp   Name
private         String     sorte;
private         String     hersteller;
private         int        bestand;
```

Auf die Datenfelder sollte nicht von außen nicht zugegriffen werden können, also werden sie üblicherweise als private (verborgen) gekennzeichnet. Wenn sie als public gekennzeichnet sind, kann von außen auf sie zugegriffen werden.

Konstruktoren

Ein Konstruktor sorgt dafür, dass ein Objekt nach der Erzeugung in einen gültigen Anfangszustand versetzt („initialisiert") wird. Ein Konstruktor ist immer public und hat denselben Namen wie die Klasse:

```
Sichtbarkeit  Name  Parameterliste
public        Apfel ()
{
   sorte = "Boskop";
   hersteller = "FruchtCoop GmbH";
   bestand = 265;
}
```

Dieser Konstruktor hat einen entscheidenden Nachteil: Er erzeugt immer Apfel-Objekte der Sorte „Boskop" vom Hersteller „FruchtCoop GmbH" mit einem Bestand von 265 Stück. Ein erweiterter Konstruktor mit Parametern ermöglicht es, dem Objekt bei der Erzeugung Attributwerte zuzuweisen:

```
public Apfel(String psorte, String phersteller, int pbestand)
{
   sorte = psorte;
   hersteller = phersteller;
   bestand = pbestand;
}
```

Eine Klassendefinition kann mehrere Konstruktoren enthalten. Für die Erzeugung eines Objektes wählt man einen geeigneten aus.

Methoden
Eine Methodendefinition besteht aus dem Methodenkopf *(Signatur)* und dem Rumpf mit den Anweisungen der Methode, z.B.:

```
Sichtbarkeit  Rückgabetyp Name        Parameterliste
public        String      getSorte    ()
{
  return sorte;
}
```

Die Methode `getSorte` gibt den Wert des Datenfeldes `sorte` als `String` zurück. `getSorte` hat deshalb den Rückgabetyp `String`. Die Methode benötigt keine Parameter. Ihre Parameterliste ist leer.

Die Rückgabeanweisung `return` beendet eine Methode und gibt den gewünschten Wert zurück. Sie sollte deshalb immer die letzte Anweisung der Methode sein.

`getSorte` ist eine *sondierende* Methode, d.h. sie fragt den Wert eines Datenfeldes ab. Eine *verändernde* Methode dagegen ändert den Wert eines Datenfeldes:

```
public void setSorte(String psorte)
{
  sorte = psorte;
}
```

Die Methode `setSorte` weist dem Datenfeld `sorte` den Wert des Parameters `psorte` zu. Da sie keinen Wert zurück gibt, ist ihr Rückgabetyp `void` („leer").

Methodennamen beginnen immer mit einem Kleinbuchstaben. Methoden, die der Kommunikation eines Objektes mit seiner Umwelt dienen, sind öffentlich (`public`). Methoden, die mit `private` gekennzeichnet sind, können nur innerhalb der Klasse aufgerufen werden.

Klassendiagramme

Datenfelder, Konstruktoren, Methoden und ihre Sichtbarkeit können in einem Klassendiagramm übersichtlich dargestellt werden: