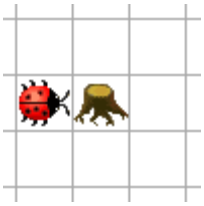
Methoden-Expertin

Aufgabe 1

Kara steht vor einem Baum, der alleine in der Welt steht. Hinter dem Baum hat es ein Kleeblatt, das Kara aufheben soll. Danach soll Kara wieder zum Ausgangsort zurückkehren.

Situation vor dem Start des Programms

Situation nachher



Das Neue

Kara kann gewisse Abläufe lernen und unter einem neuen Kommando speichern. Diese Kommandos werden in Java Methoden genannt. Kara lernt in dieser Aufgabe, was er bei `geheUmBaumHerum()` und bei `dreheUm180Grad()` machen muss.

Lösung in JavaKara

```
import JavaKaraProgram;
public class GeheUmBaumHerumUndNimmKleeblattAuf extends
JavaKaraProgram
{ // Anfang von GeheUmBaumHerumUndNimmKleeblattAuf
  void geheUmBaumHerum() // Methodenkopf
  {
    kara.turnLeft();
    kara.move();
    kara.turnRight();
    kara.move();
    kara.move();
    kara.turnRight();
    kara.move();
    kara.turnLeft();
  }

  void dreheUm180Grad() // Methodenkopf
  {
    kara.turnRight();
    kara.turnRight();
  }
```

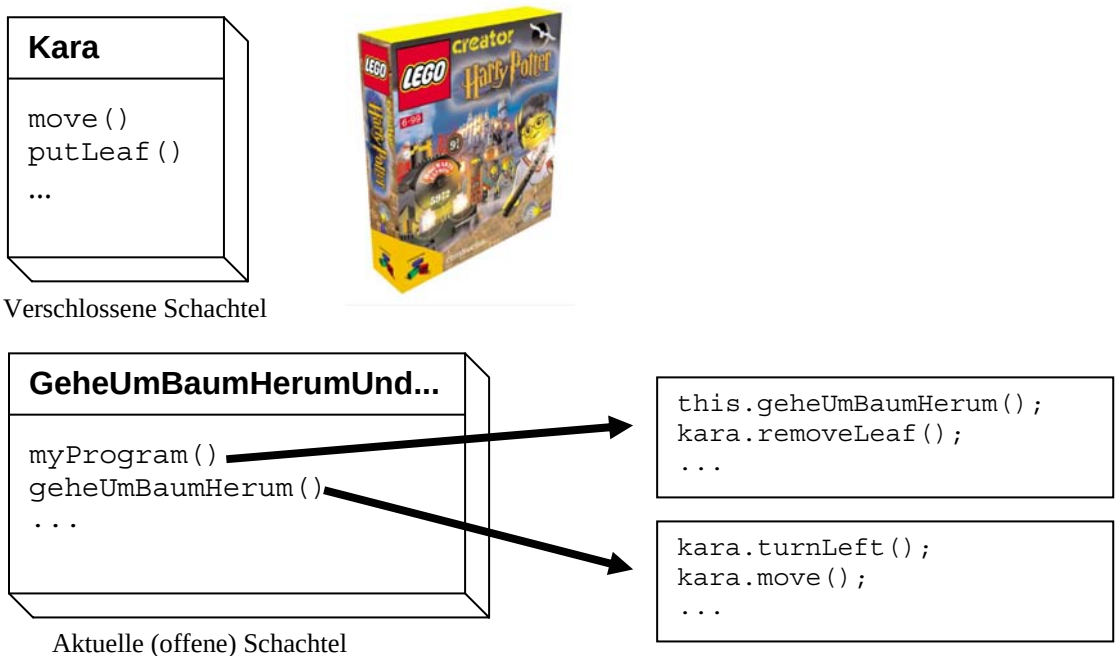
```
  public void myProgram()
  { // Anfang von myProgramm
    this.geheUmBaumHerum(); // Methodenaufruf
    kara.removeLeaf();
    this.dreheUm180Grad();  // Methodenaufruf
    this.geheUmBaumHerum(); // Methodenaufruf
    this.dreheUm180Grad();  // Methodenaufruf
  } // Ende von myProgramm
} // Ende von GeheUmBaumHerumUndNimmKleeblattAuf
```

Erläuterungen

1. **Der Aufruf der Methode**
   Mit `this.geheUmBaumHerum();` wird Kara mitgeteilt, dass er die Methode mit dem Namen geheUmBaumHerum ausführen soll. Das `this` vor dem Methodennamen sagt Kara, dass sich die Methode in der aktuellen Klasse befindet. Dies ist aber nicht unbedingt nötig. Möglich wäre auch:

   ```
   (...)
   geheUmBaumHerum();
   (...)
   ```

   Uff, das tönt alles ein bisschen abstrakt! Du musst dir das einfach so vorstellen:



   Die Anleitung `geheUmBaumHerum()` steckt nicht in der Kara-Schachtel, sondern in der aktuellen Schachtel. Erst die Anleitung ruft einen Befehl aus der Kara-Schachtel auf.

2. **Das leere Klammerpaar**
   Das Klammerpaar am Schluss des Methodenaufrufs bzw. des Methodenkopfs bedeutet, dass man keine Parameter übergeben möchte. Mit Hilfe von Parametern könnte man dem Kara z.B. mitteilen, wie viele Bäume hintereinander stehen, um die er herumgehen soll. Doch dazu kommen wir später wieder.

3. **Die Methode**
   Beim Methodenkopf wird dem Namen das Schlüsselwort `void` vorangestellt. Dieses Schlüsselwort gibt an, was für einen Wert wir an das Hauptprogramm zurückgeben wollen. Void bedeutet im Englischen „leer“, wir möchten also keinen Wert zurückgeben.
   Aus dem Mathematik-Unterricht kennen wir aber viele Beispiele, wo wir einen Wert zurückbekommen: Der Methode „Sinus“ übergeben wir einen Wert (z.B. $\pi/2$) und wir bekommen die Zahl 1 zurück. Auch darauf werden wir erst später eingehen.

Aufgabe 2

Nun soll Kara Rechtecke beliebiger Grösse zeichnen können. Dafür wäre es nützlich, eine Methode zu haben, mit der Kara eine Seite zeichnen kann, d.h. mehrere Kleeblätter hintereinander legen kann.

Das Neue

In einer neuen Methode `legeKleeblaetter` soll Kara die Kleeblätter legen. Die Anzahl der zu legenden Blätter wird mit einem Parameter übergeben.

Lösung in JavaKara

```
import JavaKaraProgram;
public class ZeichneRechteck extends JavaKaraProgram
{ // Anfang von ZeichneRechteck
  void legeKleeblaetter(int anzahl)
  {
    for (int i=1; i<=anzahl; i++)
    {
      kara.putLeaf();
      kara.move();
    }
  }
  public void myProgram()
  { // Anfang von myProgramm
    // 6x4 Rechteck zeichnen...
    legeKleeblaetter(5); // 5 Kleeblätter legen
    kara.turnRight();
    legeKleeblaetter(3); // 3 Kleeblätter legen
    kara.turnRight();
    legeKleeblaetter(5); // 5 Kleeblätter legen
    kara.turnRight();
    legeKleeblaetter(3); // 3 Kleeblätter legen
  } // Ende von myProgramm
} // Ende von ZeichneRechteck
```

Bemerkung

Die Anweisung `for (int i=1; i<=anzahl; i++)` sorgt dafür, dass die beiden Anweisungen `kara.putLeaf()` und `kara.move()` nicht nur einmal, sondern *anzahl*-mal ausgeführt werden.

Erläuterungen

Bis jetzt sind immer nur Methoden vorgekommen, denen keine Parameter übergeben worden sind. Das waren nur Spezialfälle, darum waren die Klammern auch immer leer. Es können aber auch mehrere Parameter übergeben werden, die durch Kommas voneinander getrennt werden:

```
void zeichneRechteck(int breite, int hoehe)
```

Für jeden Parameter, den man einer Methode „mitgibt“, muss man seinen Datentyp angeben. Für die Anzahl der Kleeblätter haben wir den Typ `int` angegeben, der für integer steht. Integer sind ganze Zahlen. Wir werden in dieser Woche aber auch noch anderen Datentypen begegnen:
`double`: Das sind Fliesskommazahlen, wie z.B. 54013.35 oder 3.14159265358979
`String`: Dort können Zeichenketten gespeichert werden, also z.B. „Kara ist super!“
Weitere Typen sind `boolean`, `Graphics` und noch viele andere mehr. Ihnen werden wir teilweise später wieder begegnen.

Aufgabe 3

Kara soll alle Kleeblätter bis zum nächsten Baum einsammeln. Kara möchte die Kleeblätter zählen und danach anzeigen, wie viele er aufgenommen hat.

Das Neue

Eine Methode `zaehleKleeblaetterBisBaum` gibt das Ergebnis des Zählvorgangs zurück.

Lösung in JavaKara

```
import JavaKaraProgram;
public class ZaehleKleeblaetter extends JavaKaraProgram
{ // Anfang von ZaehleKleeblaettter
  int zaehleKleeblaetterBisBaum()
  {
    int kleeblattZaehler = 0;
    while (!kara.treeFront())
    {
      kara.move();
      if (kara.onLeaf())
      {
        kara.removeLeaf();
        kleeblattZaehler++;
      }
    }
    return kleeblattZaehler;  // Wert zurueckgeben
  }

  public void myProgram()
  { // Anfang von myProgram
    int zaehler = zaehleKleeblaetterBisBaum();
    tools.showMessage("Ich habe "+zaehler+
                      " Kleeblaetter gefunden.");
  } // Ende von myProgram
} // Ende von ZaehleKleeblaettter
```

Bemerkung

Die Anweisung `while (!kara.treeFront())` sorgt dafür, dass Kara die folgenden Anweisungen solange ausführt, bis er vor einem Baum steht.

Erläuterungen

Bisher mussten wir immer `void` vor unsere Methoden stellen. Jetzt sehen wir, was dies genau bedeutet. Für jede Methode gibt man an, was für einen Typ der Rückgabewert hat. Hat eine Methode keinen Rückgabewert, so schreibt man `void`. Diese Methoden sind also eigentlich Spezialfälle.
Die Rückgabe eines Wertes wird mit dem Aufruf von `return kleeblattZaehler;` gemacht.

Mit `tools.showMessage` kann ein Text (vom Typ `String`) ausgegeben werden. Mit `+` werden dabei verschiedene Texte aneinandergehängt.

Fragen

1. **Rechteck-Methode**
   Kreiere eine Methode `void zeichneRechteck(int breite, int hoehe)`, die mit Kleeblättern ein Rechteck zeichnet. Zeichne damit ein Rechteck der Grösse 4x3.
   Hinweis: Benutze das JavaKara Programm von Aufgabe 2 als Hilfe.
   Teste das Programm am Computer.

2. Was passiert, wenn man beim Rechteck eine Grösse von 1x3 zeichnet? Wie könnte man das Problem beseitigen?
   Probiere deine Lösung am Computer aus!

3. *Zusatzaufgabe:* Wir möchten nun die Aufgabe 3 so verändern, dass Kara nicht bis zum nächsten Baum läuft, sondern nur eine bestimmte Anzahl Schritte macht. Dabei soll er wie gehabt die aufgenommenen Kleeblätter zählen. Der Methodenkopf soll also wie folgt aussehen:
   `int zaehleKleeblaetter(int anzahlSchritte)`
   Entwirf die Methode auf Papier und teste sie anschliessend am Computer.