Erste Schritte mit Java

Grundlagen

Um ein Java-Programm zu erstellen, muss man zunächst seinen **Quelltext** schreiben. Das ist eine Testdatei mit der Dateierdung .java, speziellen Befehlen und dem folgenden prinzipiellen Aufbau:

```
public class <<Dateiname>>{  
    << Allgemeine (Globale) Definitionen >>  
  
    public static void main(String args[]){  
        << Hier beginnt das eigentliche Programm >>  
    }  
}
```

Wichtig ist das, dass <<Dateiname>> mit dem eigentlichen Dateinamen (abgesehen von der Endung .java) identisch ist. Eine genaue Erklärung des Aufbaus ist leider im Moment nicht möglich. Deshalb muss man diesen Rumpf einfach nur als Vorlage verwenden, ohne ihn zu verstehen.

Java-Quelltexte sind normale Textdateien und können damit mit jedem beliebigen Texteditor erstellt werden. Als sinnvoll haben sich aber Editoren erwiesen, die die Java-Befehle hervorheben, da man dann besser im Quelltext arbeiten kann.

Um einen Java-Quelltext in ein Programm umzuwandeln, muss ihn **kompilieren**. Dies geschieht mit dem Programm „javac.exe“. Es erzeugt aus der Textdatei eine Binärdatei mit der Dateierdung .class, die das eigentliche Java-Programm darstellt.

Da aber Java plattformunabhängig ist, kann man das Programm nicht direkt ausführen. Man braucht eine sogenannte Virtual Machine, die das Programm dann so umsetzt, das der Computer es ausführen kann. Zum Ausführen des Java-Programms, und damit zum Starten der Virtual Machine, verwendet man das Programm „java.exe“.

Für Java geeignete Texteditoren enthalten diesen Programmaufrufe normalerweise in ihrem Menü oder als Symbol.

Das erste Java-Programm

```
public class HalloWelt{  
  
    public static void main(String args[]){  
        System.out.println("Hallo Welt!");  
    }  
}
```

An diesem Programm erkennt man schon einige wichtige Sachverhalte:

- Jeder Befehl muss in Java mit einem Strichpunkt ; enden.
- Um einen Text auszugeben, kann man die Funktion/den Befehl (in Java Methode genannt) System.out.println verwenden.
- Den Text, den man ausgeben will, schreibt man innerhalb der Klammern in Anführungszeichen.
- Java erwartet nicht einen Befehl pro Zeile. Trotzdem sollte man seine Quelltext sinnvoll strukturieren. Eine solche Möglichkeit ist im Beispiel dargestellt. Wie so oft gilt also auch hier: *Weniger* (Ausdrücke pro Zeile) *ist mehr* (Übersichtlichkeit).

Aufgabe:

Tippe den Quelltext in einem Texteditor ab und speichere ihn unter dem Namen „HalloWelt.java“. Kompiliere den Quelltext und führe das Programm aus.

Erste Berechnung in Java:

```
public class ErsteBerechnung{  
  
    public static void main(String args[]){  
        int i;  
        i=3+4;  
        System.out.println(i);  
    }  
}
```

Interessant ist der Ausdruck `int i`; Hier wird eine **Variable** namens `i` deklariert. Man sagt auch `i` ist der Bezeichner der Variablen.

Variablen sind Behälter, in denen man verschiedene Werte speichern kann. Im Gegensatz zur Mathematik muss man in Java sofort angeben, welche Werte man überhaupt in der Variable ablegen will. Dies geschieht durch das `int`, was angibt, dass nur ganze Zahlen in der Variablen `i` gespeichert werden sollen. „int“ bezeichnet man als **Datentyp**. Andere Datentypen sind z.B. **double** für Dezimalzahlen („Fließkommazahlen“) oder **char** für einzelne Zeichen.

Ein Bezeichner setzt sich in Java aus folgenden Elementen zusammen:

- den *Buchstaben* des Alphabets (d.h. a,b,c,...). Java unterscheidet hierbei zwischen Groß- und Kleinschreibung.
- dem *Unterstrich* „_“
- dem *Dollarzeichen* „\$“
- den *Ziffern* 0, 1, 2, ..., 9

Der nächste Ausdruck (`i=3+4`;) ist eine **Zuweisung**. Die Zahlen 3 und 4 auf der rechten Seite werden addiert und das Ergebnis wird in der Variablen `i` gespeichert.

Wichtig bei einer Zuweisung ist, dass immer zunächst der Ausdruck auf der rechten Seite berechnet wird und erst dann das Ergebnis. Auf der rechten Seite können auch Variablen stehen, sogar die Variable von der linken Seite. Wenn eine Variable `x` den Wert 5 hat, dann enthält sie nach dem Ausdruck `x=x+3`; den Wert 8.

Mittels `System.out.println(i)`; wird der Inhalt der Variablen `i` auf dem Bildschirm ausgegeben. Wenn man der Ausgabe keine neue Zeile beginnen will, verwendet man stattdessen `System.out.print(i)`;

Aufgaben:

1. Welche der folgenden Bezeichner sind in Java erlaubt? Warum sind sie die anderen nicht erlaubt?
Hallo_Welt, 10kleineNegerlein, _H_A_L_L_O_, Das_war's, hallo123, hallo_123, Hallo Welt, class
Die gleichen Regeln wie bei Bezeichnern von Variablen gelten auch für die Dateinamen!
2. Schreibe ein Java-Programm, dass die Zahlen 12 und 9 addiert, subtrahiert, multipliziert und dividiert. Dabei soll das jeweilige Ergebnis in einer passenden Variablen gespeichert und diese ausgegeben werden.