

II.1.1 Erste Schritte

Freitag, 21. Oktober 2016 17:00

Definiere nun die Syntax von Java mit Hilfe von Syntaxdiagrammen.

Komplette Grammatik-Def.
für Java: Java Language
Specification (Webseiten von Oracle)

Java-Prog.: Sammlung von Klassendeklarationen

In Java: Damit ein Prog. ausgeführt werden kann,
muss es in der entspr. Klasse eine Methode `main`
geben, mit Eingabeargument v. Typ `String[]`
und mit Ausgabotyp `void`.

Variablen deklaration

(z.B. `int x, y;`)

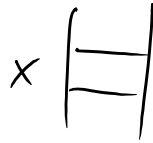
Vereinbart 2 Variablen `x, y`
und legt ihren Typ fest
(`int`, d.h., sie können mit
ganzer Zahl belegt werden).

Jede Var. muss vor Verwendg.
deklariert werden.

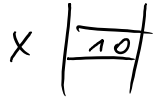
Variablen dekl. stellt Speicherplatz entsprechender Größe zur

Verfügung.

int x;



x = 10;



Java-Ausführung:

- Jede Datei darf höchstens eine "public"-Klasse enthalten.
(z.B. Rechnung.java)

Konstanten

final: Var. kann einmal einen Wert zugewiesen bekommen, aber Wert kann danach nicht mehr geändert werden.

Auch möglich:

```
final int x, y;
```

⋮

```
x = 10;
```

```
y = ...;
```

Eingabe + Import von Paketen

- Rechnung verwendet Typ (Klasse) Scanner, die in einem Paket java.util definiert ist.
- Durch `import java.util.Scanner;` am Anfang der Datei ist die Klasse Scanner danach überall verwendbar.
- Durch `import java.util.*;` werden alle Klassen des Pakets java.util importiert.
- Kommentare in Programmen:
 - nötig für Verständlichkeit !!
 - // bis Zeilenende
 - /*
:
*/
- `sc` ist Var. vom Typ Scanner. Ihr wird neues Objekt vom Typ Scanner zugewiesen. Dieses neue Objekt liest Eingaben von der Tastatur (System.in).
- Objekt `sc` stellt mehrere Methoden zur Verfügung.

z.B. ist `sc.nextInt()` eine Methode, die die nächste integer-Zahl von der Tastatur einliest.

Polymorphismus

Ein Funktionssymbol (wie `+`)

kann auf Argumente versch. Typen angewendet werden.

Hier: Funktionsweise von `+` ist unterschiedlich für Argumente verschiedenen Typs.

`2 + 3` ist `5` (Addition)

`"hal" + "lo"` ist `"hallo"` (Stringkonkatenation)

`"hal" + 2` ist `"hal2"`

`2 + 3 + "hal"` ist `"5hal"`

`"hal" + 2 + 3` ist `"hal23"`

Auch `System.out.print` ist polymorph, d.h., die Funktion kann ebenfalls für Argumente versch. Typen aufgerufen werden.

Dann werden diese Argumente mit einer Methode `toString` in Strings umgewandelt.

Methodenaufrufe

z.B. `System.out.print(x+y);`

d.h. als Anweisung

Methodenaufrufe können auch in Ausdrücken verwendet werden.

Bsp.: verwendet Methode `max` aus der vordef. Klasse `Math`.

Java Development Kit (JDK)

enthält große Bibliothek definierter Klassen.

Diese Klassen sind in der entspr. Dokumentation beschrieben
(Application Programmer Interface, API).

Bedingte Ausdrücke

$Aus_1 ? Aus_2 : Aus_3$

wertet zu Aus_2 aus, falls Aus_1 zu
true auswertet

wertet zu Aus_3 aus, falls Aus_1 zu false auswertet

Bspe

• Grundwerte: 1, 2, true, "hallo", ...

d.h.: Werte von grundlegenden Datentypen, siehe Abschn. II 1.2

• Name: `x`, `betrag`, ... d.h. Variablen

• Methodenaufruf: `Math.max(x, y)`

• Klammersetzung ist immer möglich: "hallo" + (x + y)

• Präfix-Operatoren \rightarrow - (2 + 3)

• Infix-Operatoren: \rightarrow