2 Grundlagen der Syntax

2.1 Anweisungen

Anweisungen (Befehle, Kommandos, Statements) sind Programm-Einheiten, die "etwas bewirken". In Java muss jede Anweisung durch ein Semikolon abgeschlossen werden.

```
System.out.prinln("Hallo!");
int a = 7 + 3;
```

2.2 Bezeichner

Bezeichner benennen Programmelemente (Klassen, Objekte, Methoden, Funktionen, Daten). Groß-/Kleinschreibung ist relevant (a ist **nicht** A). Bezeichner können mit einem Buchstaben, einem Unterstrich oder einem Dollarzeichen $ beginnen. Danach sind auch Zahlen zulässig.
Bezeichner sind somit von ihnen (frei) wählbare Programmelemente.

2.3 Konventionen in Java

Wie in den meisten Bereichen, gibt es auch bei der Programmierung in Java Richtlinien bzw. Konventionen, an die man sich halten sollte.

- **Klassennamen** beginnen mit einem Großbuchstaben
- **Methodennamen** beginnen mit einem Kleinbuchstaben
- **Konstanten** bestehen aus Großbuchstaben (z.B. PI)
- **Attribute/Variablen** beginnen mit einem Kleinbuchstaben
- **Paketnamen** bestehen ausschließlich aus Kleinbuchstaben

2.4 Blöcke

Blöcke werden durch geschweifte Klammern gebildet:

```
{   anweisung; ... }
```

Sie werden als eine Anweisung behandelt, können geschachtelt werden und bestimmen (auch) die Sichtbarkeit von Variablen. **Lokale Variablen** sind Variablen, die innerhalb eines Blockes deklariert werden und die auch nur in diesem Block sichtbar sind.
Klassen und Methoden bestehen aus mindestens einem Anweisungsblock.

2.5 Schlüsselwörter

Schlüsselwörter sind reservierte Wörter für Elemente der Programmiersprache Java. Diese dürfen nicht für eigene Bezeichner verwendet werden.

Tabelle mit Schlüsselwörtern

abstract	boolean	break	byte
case	char	class	const
continue	default	do	double
else	extends	finally	float
for	goto	if	implements
import	instanceof	int	long
native	new	package	private
protected	return	short	super
switch	synchronized	this	throws
transient	try	void	while

2.6 Deklarationen

Eine **Deklaration** (Vereinbarung) legt die **Attribute**, also die Eigenschaften wie *Typ, Name, Zugriffsrecht, Sichtbarkeit usw.* von Programmelementen (Variablen, Klassen, Methoden, ...) fest.

Für die Deklaration von Daten in Java gilt dann folgende Syntax:

```
[modifier] typ datum1, datum2, ...;
```

- *modifier* (Modifikatoren) legen Eigenschaften von Programmelementen fest, werden aber an dieser Stelle nicht behandelt
- *typ* legt den Typ der Daten fest. Bei *typ* kann es sich um die Angabe eines Grunddatentyps oder um eine Klasse handeln (Referenzdatentyp).

Beispiele für Deklataion:

`int i, j, k;`	Deklarationen von drei Variablen vom Typ (Grundtyp) `int`.
`String str;`	Deklarationen der Variable `str`; `str` ist ein Objekt der Klasse `String`.
`Point p1, p2;`	Deklarationen von zwei Referenzdatentypen `p1, p2` der Klasse `Point`.

Deklarationen von Daten sind in Klassen, Methoden und Programmblöcken zulässig. Die Position der Deklaration bestimmt unter anderem die Sichtbarkeit der Daten. Die einfachste Form ist die Deklaration von lokalen Variablen innerhalb eines Blockes.

2.7 Eingabe-Applikation

Die einfachste Möglichkeit, Eingaben von der Tastatur an ein Java-Programm zu übergeben, besteht darin, die Eingaben schon beim Aufruf des Programms als sogenannte Aufrufparameter anzugeben. Die Aufrufparameter werden als Folge von Zeichenketten (Zeichenfolgen, String) unter dem angegebenen Namen gespeichert, der in der Methode `main()` als String-Name angegeben ist.

In `public static void main(String[] args)` enthält `args` die Lister der beim Aufruf angegebenen Zeichenfolgen.

Beispiel:
Das Programm gibt in einer Schleife die beim Programmstart angegebene Zeichenfolgen nacheinander aus:

```
public class Eingabe1 {
  public static void main (String[] args) {
    int i = 0;
    while(i < args.length) {
      System.out.println(args[i]);
      i = i+1;
    }
  }
}
```

2.8 Ausgabe-Applikation

Alle Beispiele zu den grundlegenden Programmierelementen können anhand einer einfachen Ausgabe ausprobiert werden.
Die folgende "einfache" Applikation kann als Gerüst für solche Ausgabeschritte in Java benutzt werden:

```
public class Ausgabe1 {
  public static void main (String[] args) {
    double w;
    w = 999.99;
    System.out.println("Hallo Java " + w);
  }
}
```

An dier Stelle ist dür das Verständnis zunächst nur wichtig:

- Jede Applikation muss die Zeile `public class NameXyz {}` enthalten.
- Der Name `NameXyz` ist frei wählbar, aber wenn er geändert wird, muss auch der Name der Quelltext-Datei geändert werden.
- Geschweifte Klammern umschließen Programmteile.
- Jede Applikation erfordert die Zeile:
  `public static void main(String[] args)`
- Innerhalb der runden Kammern von `main()` muss `String` vereinbart und ein frei wählbarer Name (z.B. `args` oder `str`, `abc`, ... angegeben werden. Die eckigen Klammern dürfen sowohl bei `String` oder hinter dem Namen stehen.

Mit `System.out.println()` wird eine Zeichenfolge auf die Kommandozeile ausgegeben.

2.9 Ausdrücke

Ausdrücke sind auch in ava die kleinsten ausführbaren Einheiten innerhalb eines Programms. Ein Ausdruck besteht mindestens aus einem Operator und einem Operanden. Jeder Ausdruck hat einen Rückgabewert.

Beispiel:

`ax + ay`

Der Operator + (arithmetischer Operator) verknüpft die beiden Operanden `ax` und `ay`, der Rückgabewert ist die Summe der beiden Operaden (wenn die beiden Operanden Fließkomma- oder Ganzzahl-Datentypen sind) oder die Aneinanderreihung der Operandenwerte (wenn einer oder beide Operanden vom Datentyp `char` oder vom Referenztypen `String` sind).

Eigenschaften von Ausdrücken

- **Rückgabewert**
  Jeder Ausdruck hat einen Rückgabewert. Wert und Typ des Rückgabewertes ergeben sich aus dem Operator und den Operanden
- **Auswertungsreihenfolge**
  Bei der Auswertung der Ausdrücke gelten die bekannten Bindungs- und Assoziativitätsregeln (z.B. "Punkt vor Strichrechnung")
- **Klammerung**
  Durch Klammerung lässt sich die Auswertungsreihenfolge festlegen/verändern
- **Nebeneffekte** Es kann zu Veränderung von Variablenwerten ohne explizite Zuweisung kommen

2.10 Zuweisungen

Die Zuweisung von Werten an Variablen geschieht in Java mit dem einfachen Gleichheitszeichen. Der Wert des Ausdrucks rechts vom Gleichheitszeichen wir der Variablen links vom Gleichheitszeichen zugewiesen. Der Typ des Ausdrucks und der Typ der Variable müssen zueinander passen.

Beispiel:

```
int a;
a = 3*6;
```

2.11 Kommentare

Kommentare dienen der besseren Lebsarkeit (Dokumentation) von Programm-Quelltexten. Sie werden vom Compiler ifnoriert. Der Standard-Kommentar beginnt mit `/*` und endet mit `*/`. Er kann sich über mehrere Zeilen erstrecken. Kommentare können nicht geschachtelt werden.

Beispiel:

```
/* Dies ist ein Kommentar */
```

Zeilenkommentar

Java (und die meisten C-Compiler) kennen den Zeilenkommentar. Der Zeilenkommentar beginnt mit `//` und endet am Zeilenende (mit dem Zeilenumbruch).

Beispiel:

```
int j = 0; // Das ist ein Zeilenkommentar
```