

Informatik 1

Erste Java Programm

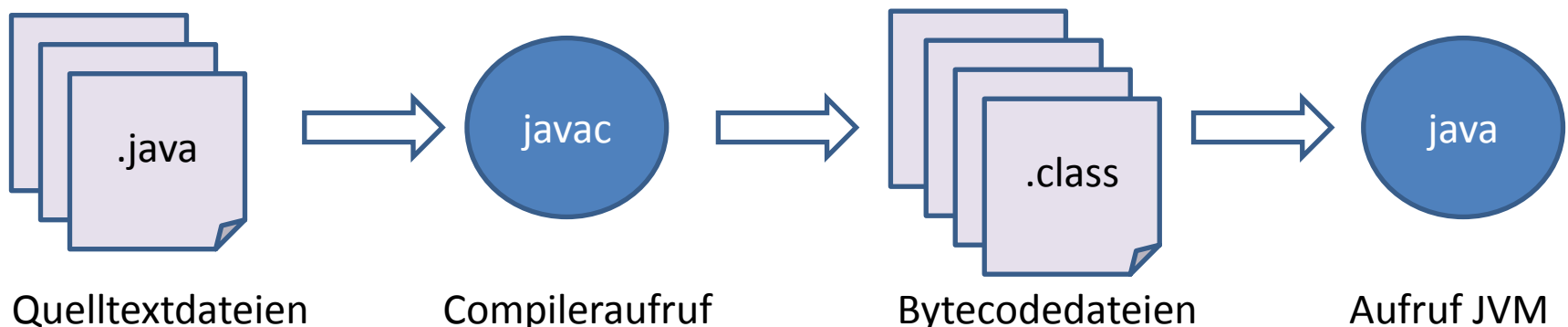
Java



- Sun Microsystems (seit 2010 zu Oracle)
- Erste Version 1995 (Projektbeginn 1991)
- James Gosling, Mike Sheridan, Patrick Naughton
- ursprünglich für interaktive Fernseher konzipiert
- objekt-orientiert, robust und sicher, weitgehend plattformunabhängig, performant, interpretiert
- "development of secure, high performance, and highly robust applications on multiple platforms in heterogeneous, distributed networks"

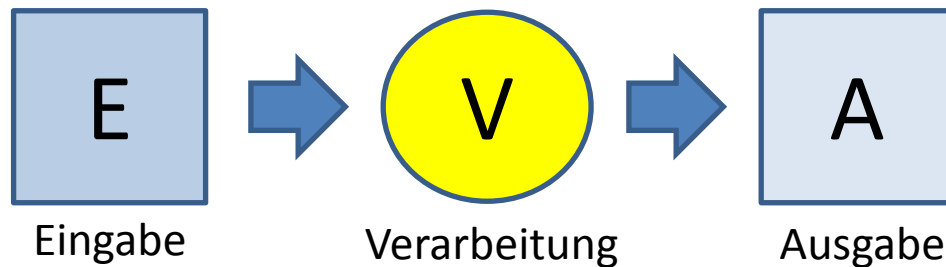
Java

- Quelltext: Textdatei, welches das Programm enthält
- Java Virtual Machine (JVM): Abstraktion eines Betriebssystems
- Compiler: erzeugt ein vom Computer ausführbares Programm
 - Normalerweise Maschinensprachprogramm
 - In Java wird Bytecode erzeugt
- Just in time compiler (JIT): JVM erzeugt Maschinensprache für die CPU während der Ausführung des Programms
- Programmierwerkzeuge: Texteditor, Compiler (javac), Interpreter (java),
- Integrierte Entwicklungsumgebung (IDE): Alle Werkzeuge unter einer GUI



Erste Java Programm

- Problem durch Eingabe und geforderte Ausgabe definieren (EVA-Prinzip)

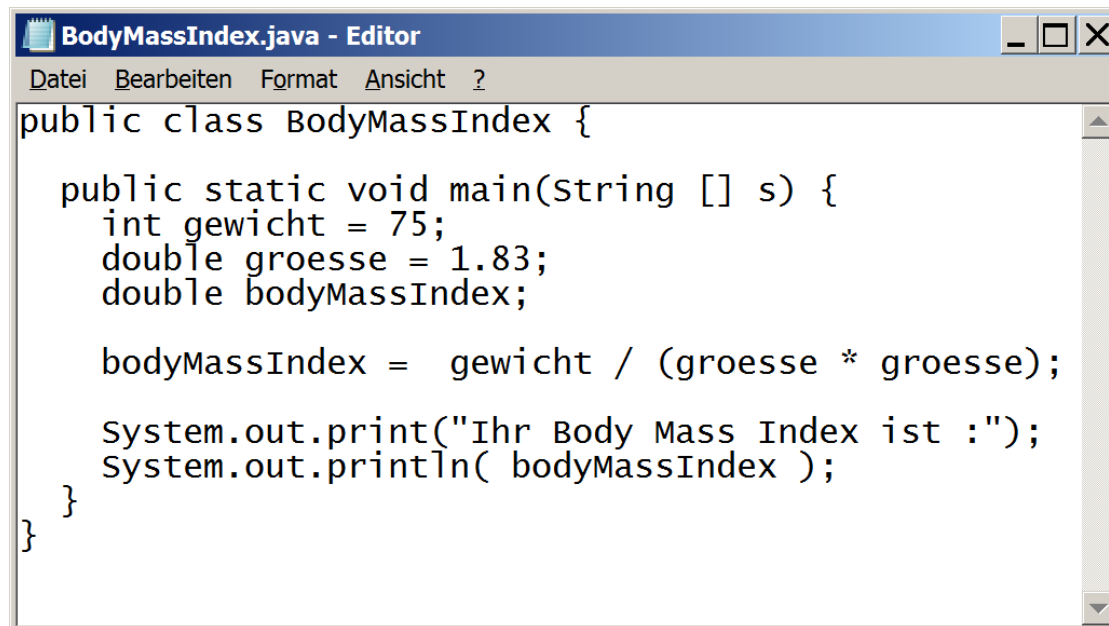


- Problem: Body-Mass-Index berechnen
 - Gegeben (Eingabe)

Gegeben:	Gewicht in Kilogramm, Körpergröße in Meter
Gesucht:	Gewicht geteilt durch Körpergröße zum Quadrat (Body-Mass-Index)

Erste Java Programm

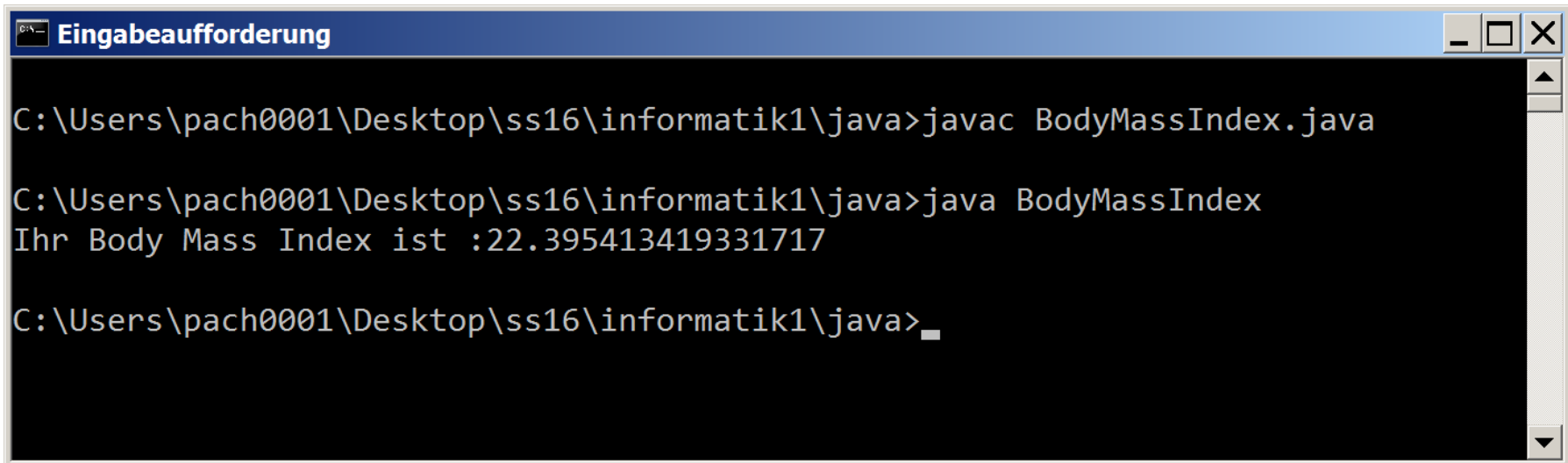
- Texteditor, z.B. Notepad.
- Zeichensatzcodierung Quelltext für javac
 - Standardcodierung des Betriebssystems
 - Unix: UTF-8, Windows: eigene



```
public class BodyMassIndex {  
    public static void main(String [] s) {  
        int gewicht = 75;  
        double groesse = 1.83;  
        double bodyMassIndex;  
  
        bodyMassIndex = gewicht / (groesse * groesse);  
  
        System.out.print("Ihr Body Mass Index ist :");  
        System.out.println( bodyMassIndex );  
    }  
}
```

Erste Java Programm

- Übersetzen mit Compiler aus der Kommandozeile (mehrere Dateien möglich)
- Ausführen des übersetzten Programms aus der Kommandozeile (ohne .class)
 - Klasse mit main-Methode muss angegeben werden



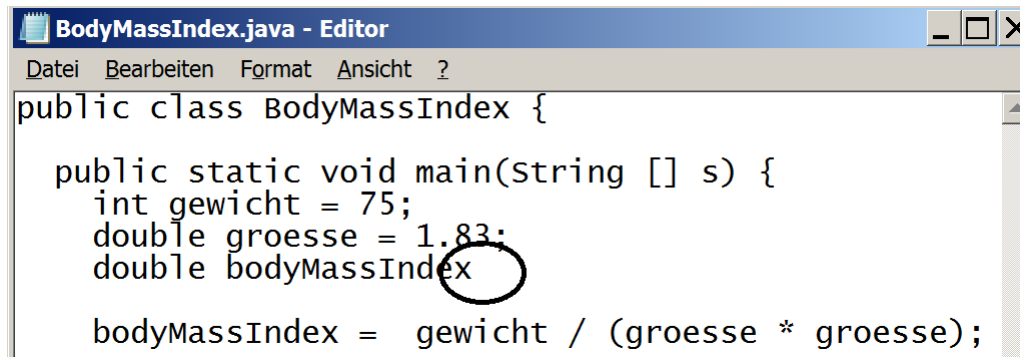
```
C:\Users\pach0001\Desktop\ss16\informatik1\java>javac BodyMassIndex.java

C:\Users\pach0001\Desktop\ss16\informatik1\java>java BodyMassIndex
Ihr Body Mass Index ist :22.395413419331717

C:\Users\pach0001\Desktop\ss16\informatik1\java>_
```

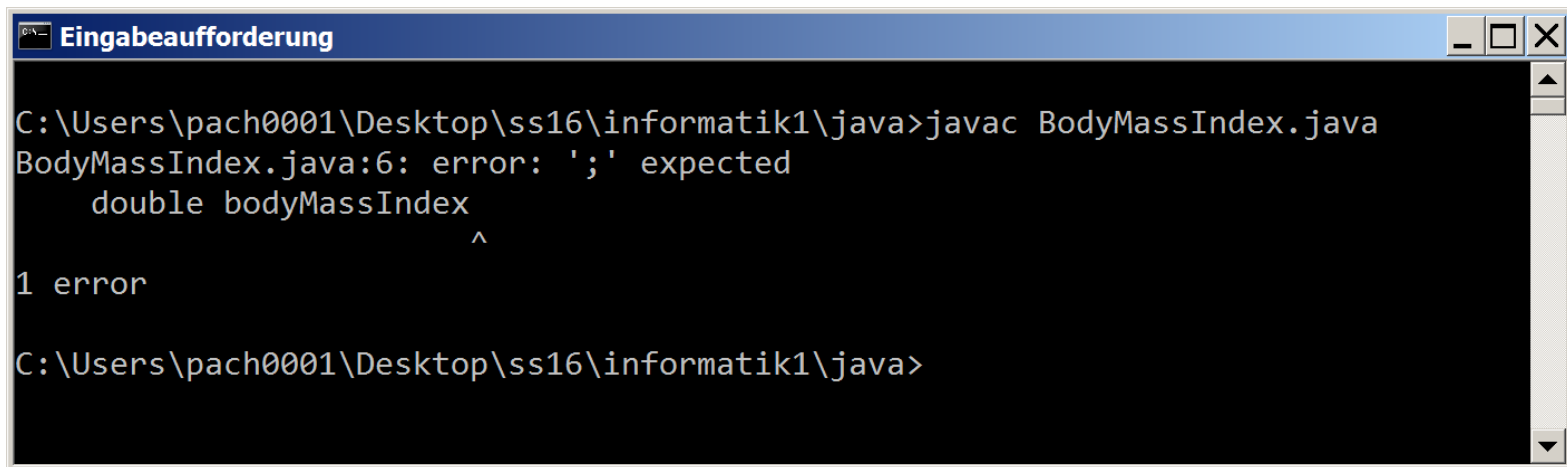
Erste Java Programm

- Syntaxfehler: Weglassen eines Semikolons



```
BodyMassIndex.java - Editor
Datei Bearbeiten Format Ansicht ?
public class BodyMassIndex {
    public static void main(String [] s) {
        int gewicht = 75;
        double groesse = 1.83;
        double bodyMassIndex
        bodyMassIndex = gewicht / (groesse * groesse);
    }
}
```

- Compiler zeigt Fehler an (auch mehrere)



```
Eingabeaufforderung
C:\Users\pach0001\Desktop\ss16\informatik1\java>javac BodyMassIndex.java
BodyMassIndex.java:6: error: ';' expected
    double bodyMassIndex
                        ^
1 error

C:\Users\pach0001\Desktop\ss16\informatik1\java>
```

Erste Java Programm

- Variablendeklaration

Führt eine Variable unter einem eindeutigen Namen ein

Variable kann Werte des Datentyps speichern

Variable kann bei Deklaration initialisiert werden

- Zuweisung

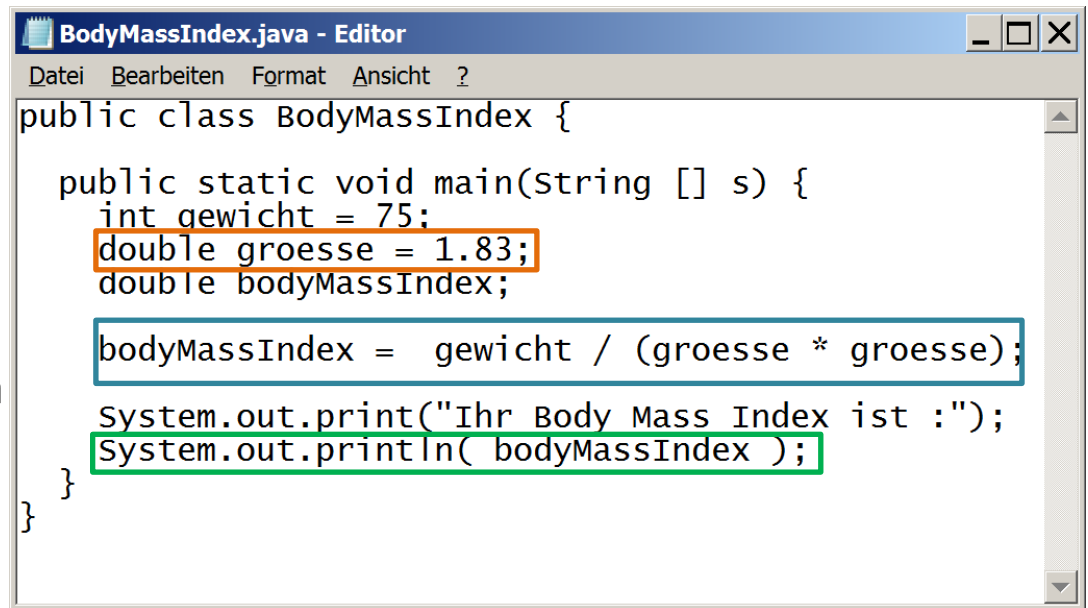
Ändert den Wert einer Variable zum berechneten Wert des Ausdrucks

- Methodenaufruf

– Ruft ein Java-Programm auf.

Es können Parameterwerte übergeben werden

- Anweisungen werden sequentiell ausgeführt



```
BodyMassIndex.java - Editor
Datei Bearbeiten Format Ansicht ?

public class BodyMassIndex {

    public static void main(String [] s) {
        int gewicht = 75;
        double groesse = 1.83;
        double bodyMassIndex;

        bodyMassIndex = gewicht / (groesse * groesse);

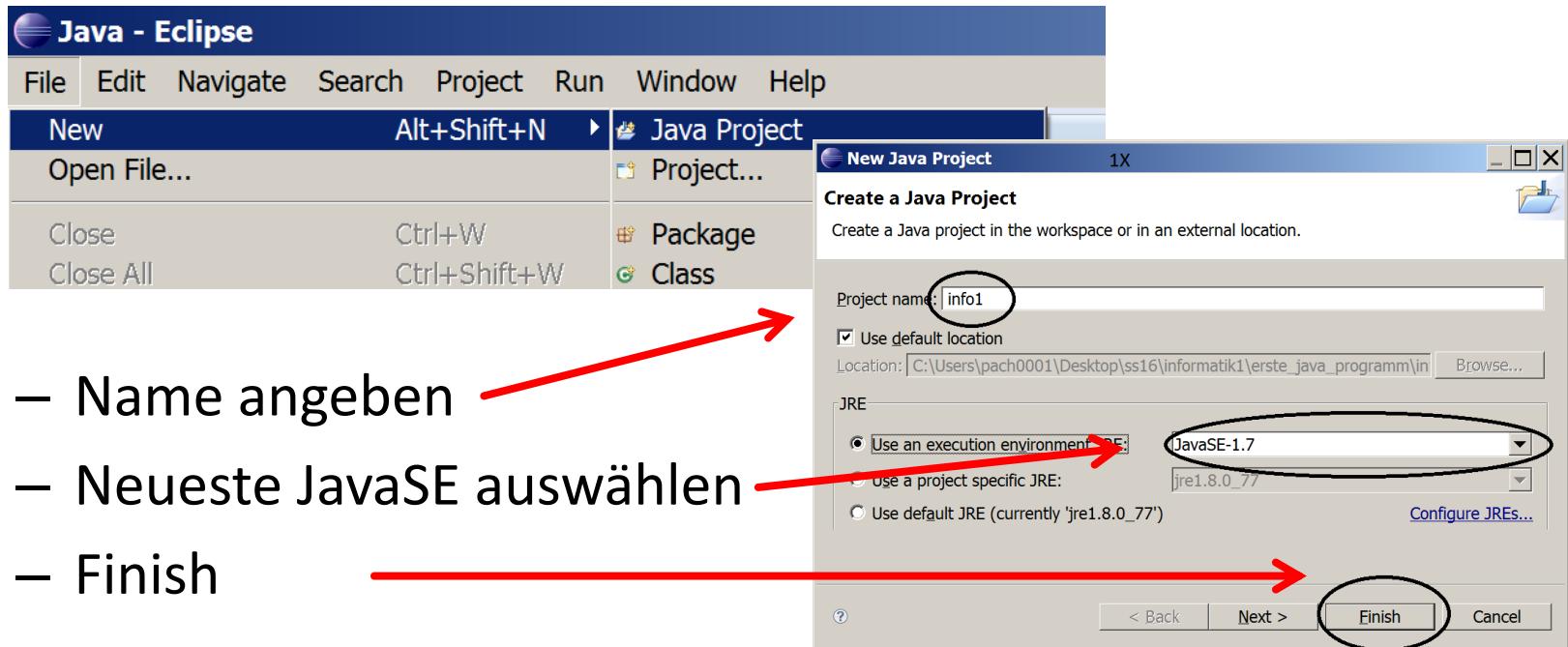
        System.out.print("Ihr Body Mass Index ist :");
        System.out.println( bodyMassIndex );
    }
}
```


Erste Java Programm

- Erste Schritte mit Eclipse
- Syntax eines Java Programms
- Datentypen
- Literale

Java Projekt erstellen

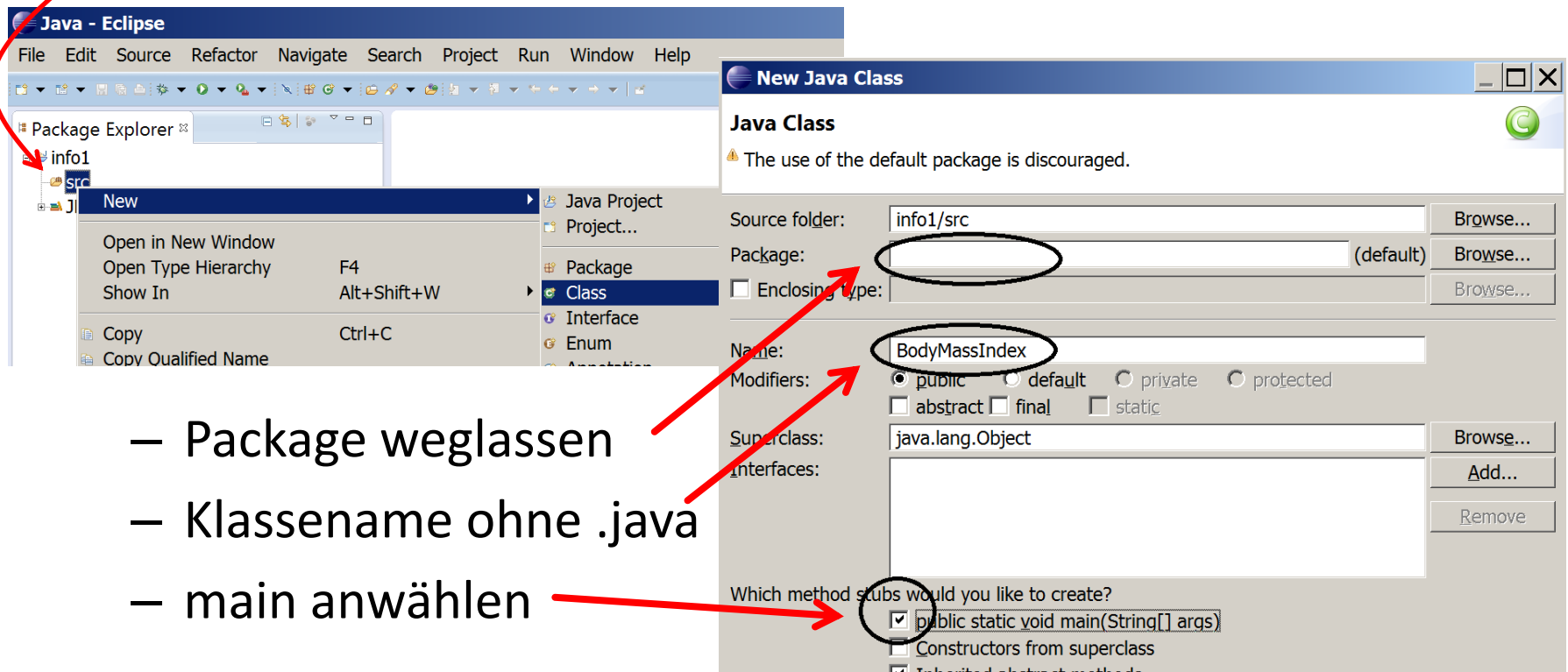
- Integrierte Entwicklungsumgebung verwenden (z.B. Eclipse, Details Rechnerübung)
 - Java Projekt erstellen (Willkommensbildschirm wegklicken)



- Name angeben
- Neueste JavaSE auswählen
- Finish

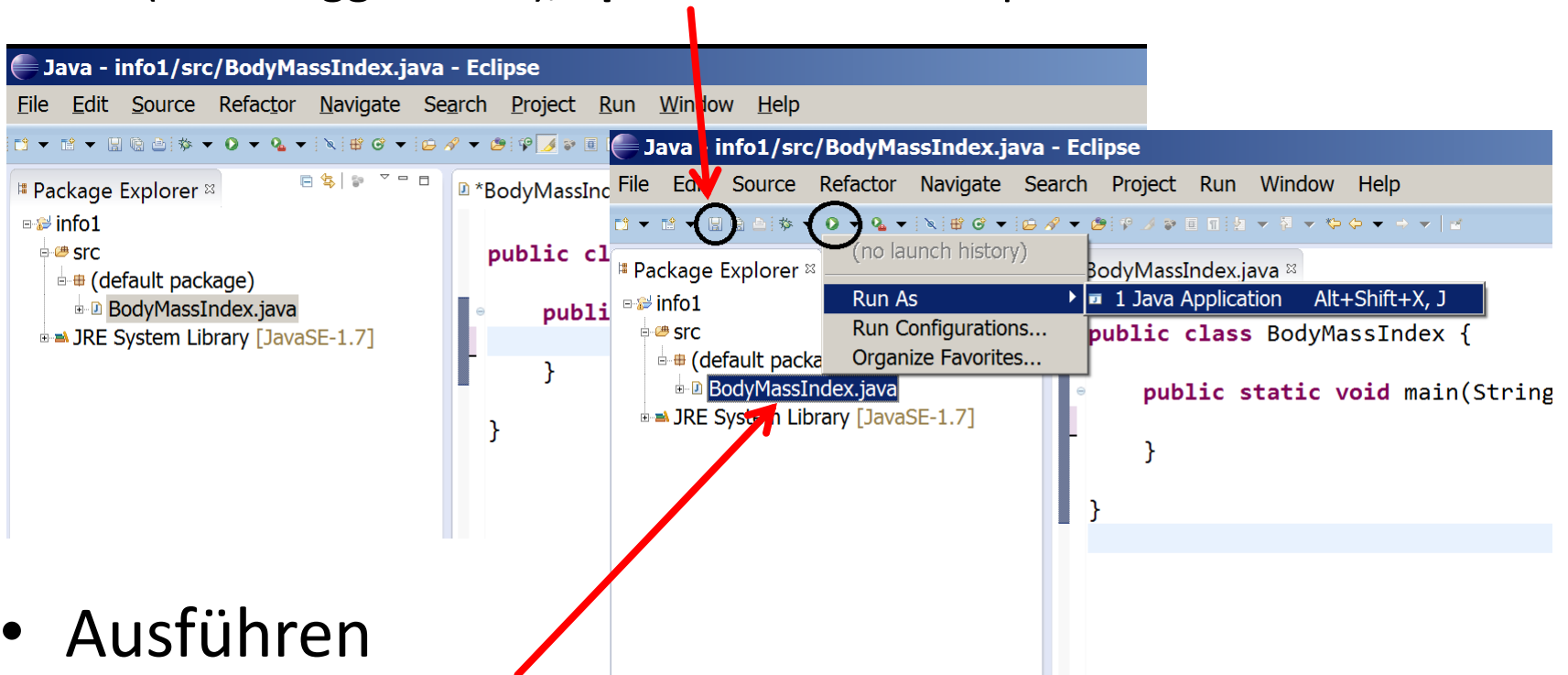
Klasse erstellen

- Klasse erstellen
 - src Anwählen, Rechtsklick



Programm ausführen

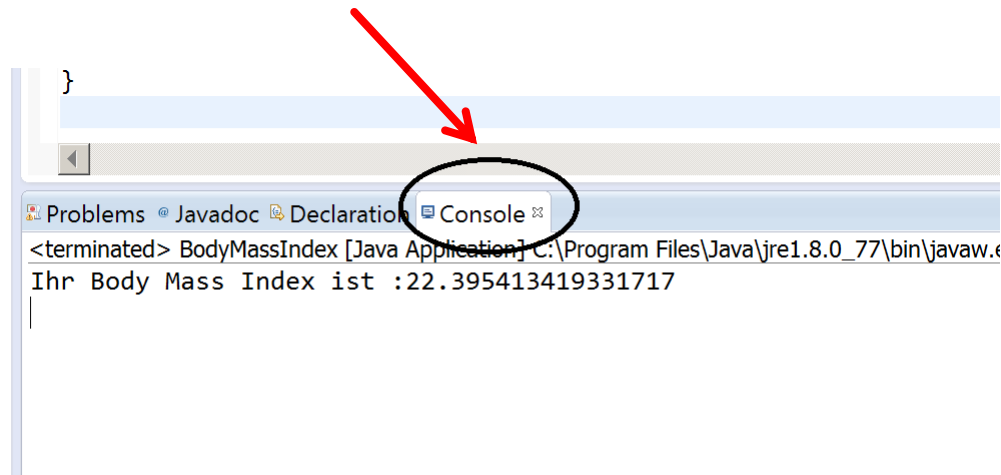
- Programm im Editorfenster schreiben
 - (hier weggelassen), **Speichern** ruft Compiler auf!



- Ausführen
 - Klasse mit main-Methode anwählen
 - mit „Run As ... -> Java Application“ starten

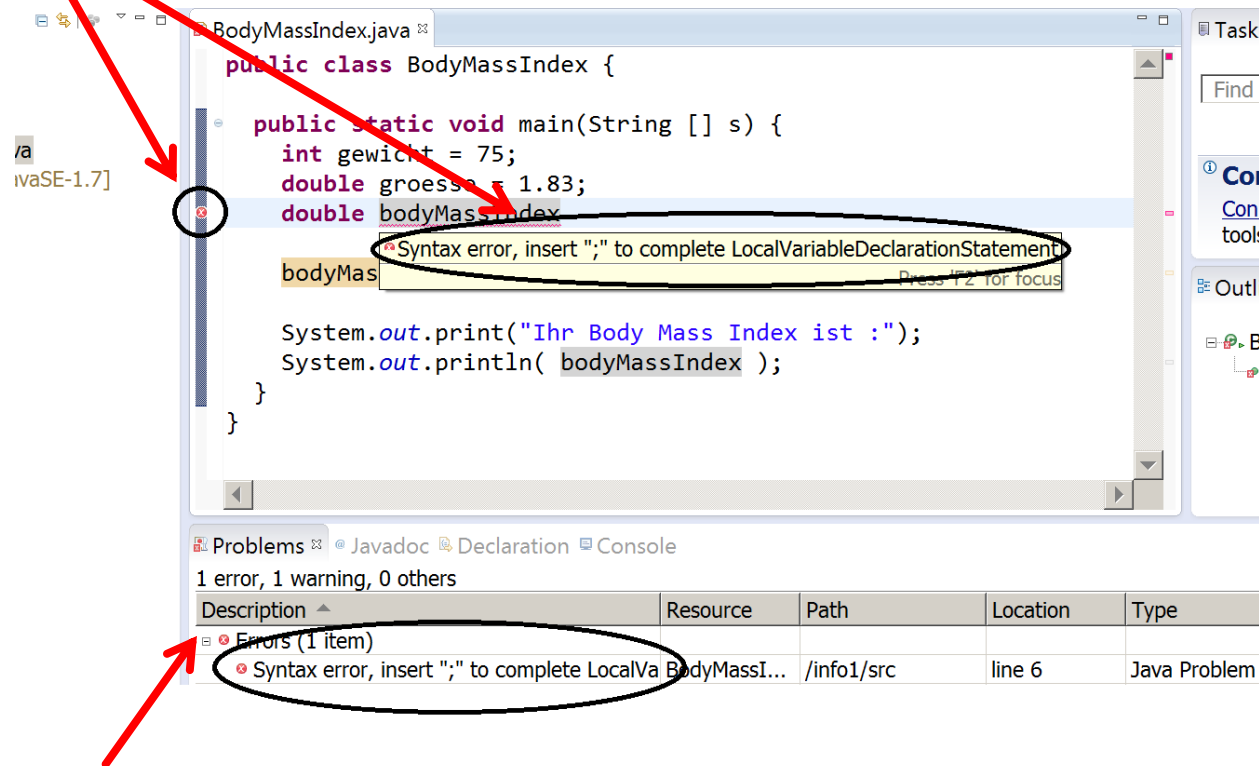
Ausgabe des Programms

- Textbasierte Ein- und **Ausgaben** in der Konsole



Quelltextfehler finden

- Syntax- und semantische Fehler
 - Im Editor, nicht immer alle Fehler, Fehlertext lesen!
(Speichern nicht vergessen)



- Unter Problems (alle Meldungen vom Compiler)

Erste Java Programm

- Syntax: Regeln, nach denen ein Programm strukturell aus Symbolen aufgebaut ist.
- Symbol: Wird aus einzelnen Zeichen im Texteditor gebildet.
- Semantik: Symbole und daraus gebildete Programmteile haben eine bestimmte Bedeutung.
- Analog zu einer natürlichen Sprache: Grammatik = Syntax, Wort = Symbol
Beispiel: "Dieser Satz kain Verb.,,
Grammatikfehler, da kein Verb vorhanden ist
Rechtschreibfehler, da "kain" falsch geschrieben ist
- Nur syntaktisch korrekte Java Programme können in ein Byte-Code-Programm übersetzt werden
- Die Syntaxfehler werden vom Compiler angezeigt
 - Die angegebene Stelle ist nicht immer die Ursache (Folgefehler)
 - Der Fehlertext bezeichnet nicht immer die richtige Ursache
 - Es ist deswegen wichtig, die Syntax und Semantik der Programmiersprache Java möglichst genau zu verstehen.

Erste Java Programm

- Vier verschiedene Symboltypen

Typ	Beispiele (erste Java-Programm)	
Schlüsselwort	int double public	Haben eine genau definierte Bedeutung
Bezeichner	BodyMassIndex main	Frei wählbare Namen
Literal	1.83 "Ihr Body Mass Index ist: "	Zur Angabe von Werten
Trennsymbol	{ } /	Alles andere

Erste Java Programm

Groß-/Kleinschreibung wird unterschieden (case sensitive)

abstract	continue	for	new	switch
assert	default	goto*	package	synchronized
boolean	do	if	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp	volatile
const*	float	native	super	while

* Nicht verwendet

Syntax Bezeichner

- Bezeichner werden vom Programmierer festgelegt
- Sie geben „Dingen“ im Programm einen Namen
- Folge von
 - Unicode-2.0 Buchstaben, z.B. griechisches Alpha, arabisches Dal, a-z
 - Dezimalziffern, 0 – 9
 - Unterstrich _
 - Dollar \$ (vermeiden!)
 - Die Folge darf nicht mit einer Ziffer beginnen
 - Schlüsselwörter dürfen nicht als Bezeichner verwendet werden
- Groß-/Kleinschreibung wird unterschieden (case sensitive)
 - *stuhl* und *Stuhl* sind verschiedene Bezeichner
- Beispiele syntaktisch gültige Bezeichner:
bezeichner \$ _ \$ _ \$ _ αLφa main
BodyMassIndex

Konventionen

Programmier**konventionen**:

- Nicht alles, was in Programmiersprachen möglich ist, sollte verwendet werden!
- Dienen der Lesbarkeit von Programmen, analog zur Groß- und Kleinschreibung sowie Leerzeichen bei der deutschen Sprache.
- Vermeiden von Programmierfehler.

Konventionen Bezeichner

Konventionen	Beispiel
Nur a-z, A-Z, 0-9 verwenden	cardriver Sportwagenfahren KFZ
_ nur in besonderen Fällen	UMFANG_ERDE
Im Wort kleinschreiben Anfangsbuchstabe von Teilwörtern groß schreiben (<u>camel style</u>)	drivingACar statt drivingacar
Keine Abkürzungen verwenden Ausnahme: Name ist üblich, da praktisch Eigenname	Kraftfahrzeug statt KFZ HTML

Trennsymbole / Kommentare

- Leerzeichen und Zeilenumbrüche sind Trennsymbole ohne Bedeutung.
- Sie werden zur Formatierung des Quelltext für den menschlichen Leser verwendet.
- Kommentare enthalten zusätzliche Information für den Leser des Quelltexts. Sie sind ebenfalls Trennsymbole.
 - Einzeilenkommentare:

```
// alles bis zum Zeilenende wird vom Compiler ignoriert
```

- Mehrzeilenkommentare (nicht verschachtelbar):

```
/* jedes Zeichen  
   bis zu den  
   letzten beiden Symbolen  
   wird ignoriert */
```

Syntax Java Programm

- Java-Programme bestehen aus einem oder mehreren Quelltexten
- Eine Quelltextdatei soll genau eine Klasse enthalten
 - Klasse kann wieder innere Klassen enthalten (Info 2)
 - Info 1: Quelltext enthält genau eine Klasse
- Die Klasse hat einen Namen
 - der Bezeichner hinter dem Schlüsselwort `class`
- Der Name der Quelltextdatei muss identisch zum Klassennamen sein + **.java**
 - `Bodymassindex.java`

Syntax Java Programm

Import- und Paketdeklarationen (später)

```
public class Klassenname {
```

Methoden- und Variablen-Deklarationen (Member)

```
}
```

Zur Übersicht alles zwischen { .. } (Block) 2 bis 4 Zeichen einrücken

- Vereinfachte und unvollständige Sicht
- Verschiedene Methoden- und Variablentypen:
 - Objekt- und Klassenmethoden
 - Objekt- und Klassenvariablen (Fields)
- Reihenfolge der Deklarationen spielt (fast) keine Rolle
- In den folgenden Beispielen:
Nur Deklaration einer Klassenmethode, der sogenannten main-Methode

Konvention Klassennamen

Konventionen	Beispiel
wie Bezeichner	
erste Buchstabe groß	Kraftfahrzeug
mindestens ein Substantiv verwenden	LegoBaukasten WasserstandBerechnen

Syntax Java Programm

- Syntax Klassenmethode am Beispiel der main-Methode (Details später)
- main ist ein Bezeichner, der den Namen der Klassenmethode definiert
- Die Namen der Klassenmethoden müssen pro Klasse eindeutig sein

```
public static class main(String [] s) {
```

Keine, eine oder mehrere **Anweisungen** (elementare Anweisungen und Kontrollanweisungen)

```
}
```

- main-Methode: wird bei Start eines Java-Programms initial aufgerufen
- Jedes ausführbare Java-Programm besitzt mindestens eine main-Methode
- Anweisung:
Kleinste vollständige Ausführungseinheit einer Programmiersprache
Stellt einen einzelnen Verarbeitungsschritt eines Programms dar

Elementare Anweisungen

- Elementare Anweisungen
 - **Deklarationen lokaler Variablen** (declaration statement)
 - **Zuweisungen** (expression statement)
 - **Methodenaufrufe** (expression statement)
 - **;** (Anweisung, die nichts tut)
- Elementare Anweisungen müssen immer mit einem **;** beendet werden
- Block **{ ... }** (gruppiert elementare Anweisungen)

Konvention Anweisungen

Konvention	Beispiel
Nur eine elementare Anweisung pro Zeile	<pre>int a = 1; double x = 7.0;</pre>
	<pre>a = (a + 1);</pre>
	<pre>System.out.print("a hat den Wert "); System.out.println(a);</pre>

Variablendeklaration

```
Datentyp Bezeichner;  
Datentyp Bezeichner = Ausdruck;
```

Initialisierung (keine Zuweisung)

- Variablen: Speichern einen Wert eines Datentyps im Hauptspeicher des Computers
- Der Datentyp beschreibt die zulässige Wertemenge und *Operationen*
- Variablen müssen vor ihrer Verwendung deklariert werden
- Lokale Variablen können nur innerhalb von Methoden deklariert werden
- Bezeichner muss eindeutig pro Methode sein
- Ausdruck muss ein Wert ergeben, der kompatibel zum Datentyp ist

Analogie Mathematik (Deklaration ohne Initialisierung)

Variablenname (Bezeichner) $\rightarrow x \in \mathbb{Z}$ \leftarrow Wertemenge (Datentyp)
 $q \in \mathbb{Q}$

$x : 2$ Ganzzahlige *Division* (Rest fällt weg)

$q : 2$ Rationale *Division*, ggf. periodische Dezimaldarstellung)

Variablendeklaration

```
int a;  
int eineVariable = 10; // Literal 10 ist ein Ausdruck  
int i = 1.0; // Fehler: 1.0 ist ein float-Literal  
           // und nicht kompatibel zu int
```

- Lokale Variablen müssen vor der ersten Verwendung mit einem Wert initialisiert werden
- Compiler prüft dies
- Lokale Variablen möglichst bei Deklaration initialisieren

```
int a = 5; // a initialisiert  
int b;  
int c;  
c = 2; // c mit Zuweisungen initialisieren  
a = c + 7;  
a = b + 1; // Fehler: b nicht initialisiert
```

Variablendeklaration

- BodyMassIndex

```
int gewicht = 75;  
double groesse = 1.83;  
double bodyMassIndex;
```



Zeitpunkt der Ausführung

Hauptspeicher (random access memory, RAM)

Initialisierten Variablenwerte befinden sich irgendwo im RAM in einer binären Codierung

gewicht

75

groesse

1.83

Konvention Variablennamen

Konventionen	Beispiele
Wie Bezeichner und:	
Erster Buchstabe klein	<code>double pi = 3.1415;</code>
Mindestens ein Substantiv verwenden	<code>int maximaleGeschwindigkeit = 120;</code>
Der Name soll gut beschreiben, was die Werte bedeuten	<code>double temperaturInGradCelsius = 22.5;</code>
Nie beschreiben, wie die Variable verwendet wird oder welchen Typ sie hat	<code>int ergebnis;</code> <code>int temp;</code> <code>int intZahl;</code>

Zuweisung

- Zuweisung: Ändert den Wert einer Variablen
- Wert wird als Ausdruck angegeben

```
Bezeichner = Ausdruck;
```

 **Zuweisungsoperator**

- Bezeichner muss der Name einer vorher deklarierten Variable sein
- Bei Ausführung der Zuweisung:
 - (linker Operand wird ausgewertet, z.B. bei Feldern)
 - Ausdruck wird schrittweise zu einem Wert ausgewertet
 - Inhalt der Variablen wird mit diesem Wert überschrieben

```
1:  int a;  
2:  a = 7;  
3:  a = a + 2;
```

Nach 1.

Nach 2.

Nach 3.

7

9

Vertauschen zweier Variablen

Gegeben:	Zwei Variablen a und b mit beliebigen initialen Werten
----------	--

Gesucht:	a und b haben ihre Werte getauscht
----------	------------------------------------

FALSCH	Richtig
<pre>int a = 7; int b = 2; a = b; b = a;</pre>	???
Skizzieren Sie den Speicher der Variablen und dessen Änderungen nach Ausführung jeder Anweisung	

Datentypen

- Acht Datentypen mit definierter binärer Codierung, Wertebereich und zugehörigen Operatoren
- 6 Zahltypen
 - Ganzzahlig: byte, short, int, double
 - Gleitkomma: float, double
- 1 Zeichen
 - char
- 1 Wahrheitswert
 - boolean

Datentypen und Werte

Typ	Standardwert	Wertebereich	Codierung	Speicher [Bit]
byte	0	-128 ... 127	8-Bit 2er-Komplement	32
short	0	-32.768 ... 32.767	16-Bit 2er-Komplement	32
int	0	-2.147.483.648 ... 2.147.483.647	32-Bit 2er-Komplement	32
long	0L	-2^{63} ... $2^{63}-1$	64-Bit 2er-Komplement	64
float	0.0f	+/- 1,4 10^{-45} ... +/- 3,4 10^{38}	32-Bit IEEE 754	32
double	0.0d	+/-4,9 10^{-324} ... +/-1,7 10^{308}	64-Bit IEEE 754	64
char	'\u0000'		16-Bit Unicode-2.0	32
boolean	false	true / false		32

Literale

- Literal: Bezeichnet Werte eines bestimmten Datentyps in Quelltexten
- Beispiel:
 - 12 Für die ganze Zahl 12 mit Datentyp int
 - 1.0 Für die Gleitkommazahl 1,0 mit Datentyp double
- Keine Literale für byte und short vorhanden
 - JVM rechnet intern ganzzahlig mit int oder long

Literale

- Ganzzahlige Literale
- Syntax (**Dezimaldarstellung**)
 - Folge von Ziffern und Unterstrichen
 - Vorzeichen + oder – möglich
 - Postfix L oder l für long
- Beispiele

1000	1_000	-42	int-Literale
1000L	1_000l	-42L	long-Literale
- Unterstrich vermeiden, L statt l verwenden
- Compiler
 - prüft Wertebereich
 - transformiert Literal in 2er-Komplement

Literale

- **Oktal Syntax**

- Folge der Ziffern 0 bis 7
- Folge beginnt mit einer 0
- Vorzeichen + / 1

- Führende 0en bei Dezimalzahlen deswegen unbedingt vermeiden!

- **Beispiele**

011	hat dezimalen Wert 9 (int)
011L	(long)

Literale

- **Hexadezimal Syntax**
 - Literal beginnt mit Präfix 0x oder 0X
 - Folge der Ziffern 0 bis 10, A, B, ... , F (a, ..., f)
 - Vorzeichen + / -
- **Beispiele**

0xFF00	0x000a	(int)
0xFFFFFFFFFL		(long)

Literale

- **Binär Syntax**

- Präfix 0b oder 0B
- Folge von 0 und 1
- Vorzeichen + / 1

- Beispiele

0b1000_000

(int)

-0b1

-1 (dezimal)

(int)

0B10L

(long)

- Gruppierung in 4er-Gruppen mit _ sinnvoll
- Zahl wird nicht als 2er-Komplement, sondern binär als Dualzahl interpretiert

Literale

- Als Programmieranfänger
 - Dezimaldarstellung verwenden
 - Führende 0 vermeiden!
 - Außer die Aufgabenstellung verlangt eine andere Darstellung

Gleitkommalliterale

- Syntax
 - Folge von Dezimalziffern
 - Nachkommanteil mit Dezimalpunkt
 - Vor- oder Nachkommaanteil optional
 - Vorzeichen + / - möglich
 - Nachgestelltes f oder F ergibt float-Literal

- Beispiele

10.5f	-14.005f	(float)
10.5	14.005	(double)
0.5	5.0	(double)
.5	5.	(double)

Gleitkommaliterale

- Wissenschaftliche Notation

- Komma wird einem Exponenten zur Basis 10 verschoben
- Exponent n wird mit e oder E direkt hinter die Zahl z geschrieben: zEn ($= z \cdot 10^n$)

- Beispiele

Dezimaler Wert

1.05e1	(double)	$1,05 \cdot 10^1$	10,5
1E5f	(float)	$1 \cdot 10^5$	10000
1.5E-2	(double)	$1,5 \cdot 10^{-2}$	0,01

- Nur bei sehr Großen oder kleinen Zahlen verwenden
- Als Anfänger vermeiden

Beispiel

Gleitkommakodierung

Gegeben	Binäre Codierung einer Gleitkommazahl nach IEEE Standard
Gesucht	Der dezimale Wert

0	10000010	00000000000000000000000000000000	10,625
---	----------	----------------------------------	--------

1	00000000	00000000000000000000000000000000	- 0
---	----------	----------------------------------	-----

- `Float.intBitsToFloat(i)` gibt für eine `int`-Wert `i`, deren Bits als Gleitkommazahl interpretiert werden, den zugehörigen `float`-Wert zurück
- **Umkehrung:** `Float.floatToIntBits(f)`
- Umwandlung `int`-Wert in Zeichenkette mit den zugehörigen binären Bits: `Integer.toBinaryString(i)`

Konvention Zahltypen

- Die JVM rechnet ganzzahlig nur mit int oder long
- CPU rechnet intern bei Gleitkommazahlen mit mehr als 64 Bits
- Konvertierungen zwischen Zahltypen können zusätzlich Ausführungszeit kosten

Konventionen	
Möglichst nur int und double verwenden	
Zahltypen möglichst nicht in Ausdrücken mischen	Int x = 3; (x + 1.0) * (x / 2) (ganzzahlige Division)
Überlauf oder Unterlauf bei int vermeiden ggf. long verwenden	

Boolesche Literale (boolean)

true	logisch wahr
false	logisch falsch

- Keine Konvertierung von oder zu Zahlen (wie in C) möglich

Zeichenliterale (char)

- Interne Codierung: 16-Bit Unicode
- Syntax
 - Einzelnes Zeichen in Hochkomma setzen oder
 - Direkte Angabe des Unicode als Hexadezimalzahl mit genau 4 Ziffern: `'\uXXXX'`
- Beispiele

<code>'0'</code>	<code>'\u0030'</code>
<code>'Å'</code>	<code>'\u01FA'</code>

Zeichenliterale

- Fluchtsymbole
 - Hochkomma kann nicht mit `“` als char-Literal angegeben werden, sondern mit Fluchtsymbol `“`
 - Damit ist auch `\` nicht mehr direkt angebbbar: `“\“`
- Weitere Fluchtsymbole
 - `“n“` Zeilenumbruch
 - `“t“` Tabulator (Cursor der Textausgabe springt zur nächsten fixen Sprungmarke)

Zeichenketten

- Datentyp String
- Kein primitiver Datentyp, aber spezielle Literale vorhanden
- Syntax
 - Folge von Zeichen in doppelten Hochkommas
- Beispiele
 - “Dies ist eine Zeichenkette”
 - “Erste Zeile\nZweite zeile\n”
- Operator + existiert für String
 - s + t ergibt Zeichenkette mit Zeichen von s gefolgt den Zeichen von t
 - einer der beiden Operanden kann primitiver Datentyp sein: der Wert wird zu einer Zeichenkette konvertiert