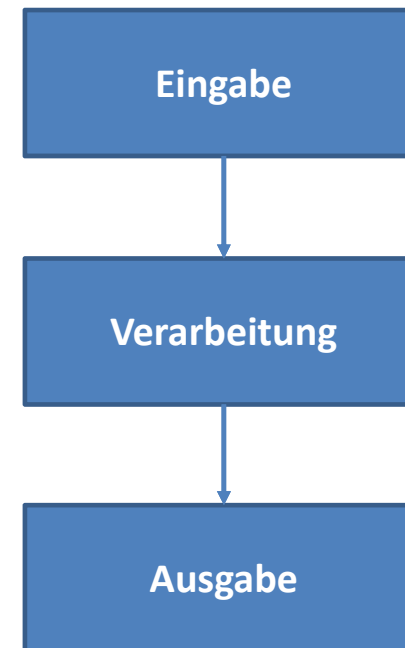


Lektion 1

EVA-Prinzip, Das erste Java-Programm
Kommentare, Ausdrücke, Variablen, Zuweisungen

EVA-Prinzip

Aufgaben eines
Computerprogramms (historisch)



Warum Java?



Wo kommt Java zum Einsatz?



source: https://images-na.ssl-images-amazon.com/images/I/71h2J6n4DNL._SL1500_.jpg

© Prof. Dr. Steffen Heinzl


Wo kommt Java zum Einsatz?



source: <http://screenshots.en.sftcdn.net/en/scrn/48000/48992/eclipse-11.jpg>

Wo kommt Java zum Einsatz?

www.deutschepost.de/de.html

Deutsche Post 

E-Post | Privatkunden | Geschäftskunden | Produkte | Service | Shop

Suche | Filiale | PLZ | Sendungsstatus

Suchbegriff >

Porto berechnen

DIN-Umschlag bis 23,5 x 12,5 cm ✓	Höhe bis 5 mm ✓	Gewicht bis 20 g ✓
---	-----------------------	--------------------------


Standardbrief national **0,58 €**

Portokalkulator

Echt stark!









Jetzt mit eigenen Briefmarken und Briefumschlägen für einen echten WOW-Effekt sorgen!

Hier bestellen



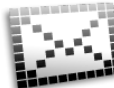
Briefmarken kaufen

Aktionsprodukte

 Internetmarke 0,45	 Postkarte 0,45
 Standardbrief 0,58	 Kompaktbrief 0,90
 Maxibrief 2,40	 Großbrief 1,45
 DHL Päckchen	 DHL Paket

Shop

Versenden | Empfangen | Schreiben | Werben





Die Post im Internet:
Der E-Postbrief.

E-Postbrief schreiben Ihre persönliche Marke	Grußkarten online Telegramm
---	--------------------------------

Produkte

- > Plusbrief individuell
- > Grußkarten
- > Mailingfactory
- > Mobil schreiben
- > Alle Produkte A-Z

Post mobil App 



Überraschen Sie mal wieder

> ... mit einem TELEGRAMM! Ihr unvergesslicher Glückwunsch von uns persönlich überbracht.

source: deutschepost.de

© Prof. Dr. Steffen Heinzl

Java EE - JSFPlatineShop/src/de/fhws/shop/hoerbuch/AudioBook.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

Project Explorer Navigator

JSFComponentIntro
JSFComposition
JSFIntro
JSFPlatineShop
JAX-WS Web Services
Deployment Descriptor: JSFPlatineShop
Java Resources
src
de.fhws.shop.beans
CartBean.java
CartItem.java
IndexBean.java
SearchBean.java
SessionBean.java
de.fhws.shop.hoerbuch
AudioBook.java
AudioBook
Libraries
Apache Tomcat v7.0 [Apache Tomcat]
EAR Libraries
JRE System Library [jdk7]
Web App Libraries
JavaScript Resources
build
WebContent
META-INF
resources
WEB-INF
lib
javafx.faces-2.1.17.jar
javax.servlet.jsp.jstl-1.2.1.jar
javax.servlet.jsp.jstl-api-1.2.1.jar
faces-config.xml

Pack-ages

Klassen

Klassenbibliotheken

```
package de.fhws.shop.hoerbuch;

import java.io.Serializable;

public class AudioBook implements Serializable
{
    String articleNumber;
    String title;
    List<String> authors = new ArrayList<String>();
    float price;
    String format;
    String isbn;
    String language;
    String imageName;
    String description;

    public AudioBook()
    {
    }

    public String getArticleNumber()
    {
        return articleNumber;
    }
    public void setArticleNumber(String articleNumber)
    {
        this.articleNumber = articleNumber;
    }
    public String getTitle()
    {
        return title;
    }
}
```

Klasse

Attribute

Konstruktor(en)

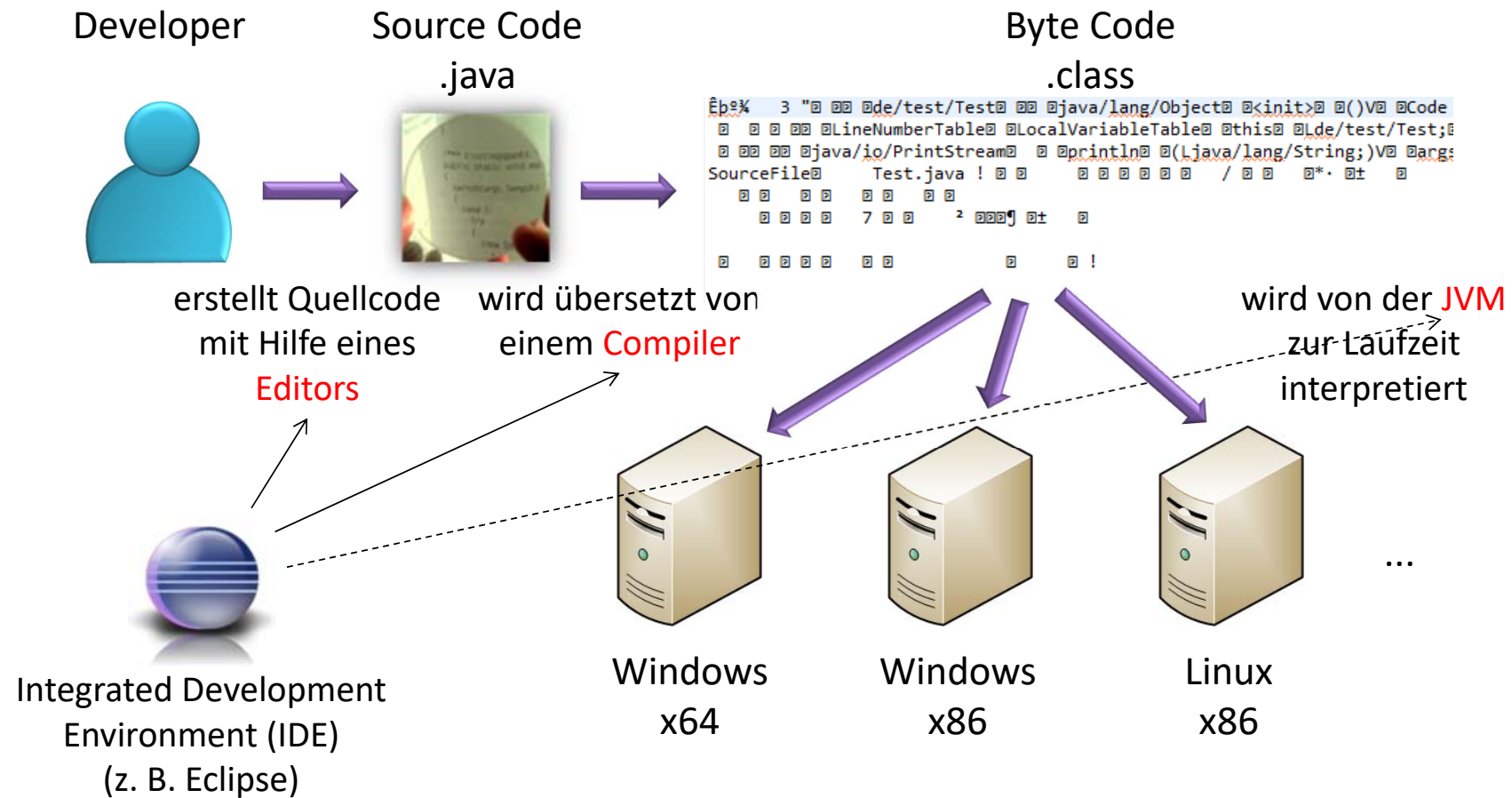
Methoden

Markers Properties Servers Data Source Explorer Snippets Console SQL Results

4 errors, 546 warnings, 7 others (Filter matched 169 of 557 items)

Das erste Java-Programm

Vom Quellcode bis zur Ausführung



Das erste Java-Programm

Vom Quellcode bis zur Ausführung

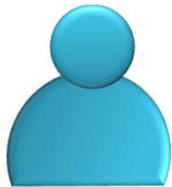
Erstellen des Quellcodes

Übersetzen in Bytecode

Ausführen des ersten Programms

Developer

Source Code
.java



erstellt Quellcode
mit Hilfe eines
Editors

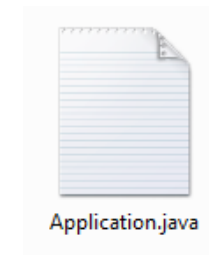
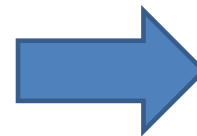
Jedes Java-Programm hat mind. eine Klasse.

Mind. eine Klasse enthält eine main-Methode.

Jede Klasse muss in einer eigenen Datei gespeichert sein.

Der Dateiname ist der Name der Klasse gefolgt von *.java*.

```
Application.java
1
2 public class Application
3 {
4     public static void main(String[] args)
5     {
6         System.out.println("Hello World");
7     }
8 }
9
```



Erstellen Sie mit einem Editor die Datei *Application.java* mit folgendem Inhalt:

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

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

Name der Klasse

Einstiegspunkt bei Programmstart

Quellcode

Befehl zur Ausgabe **auszugebender Text**

Anweisung (wird mit Semikolon abgeschlossen)

Das erste Java-Programm

Vom Quellcode bis zur Ausführung

Erstellen des Quellcodes

Übersetzen in Bytecode

Ausführen des ersten Programms

Benutzen Sie das Programm `javac` unter `JAVA_HOME/bin`,
um eine `.java` Datei zu kompilieren:

```
javac <filename>
```

In unserem Beispiel:

```
javac Application.java
```

Durch diesen Befehl wird die Datei *Application.class*
erstellt, die den Bytecode enthält.

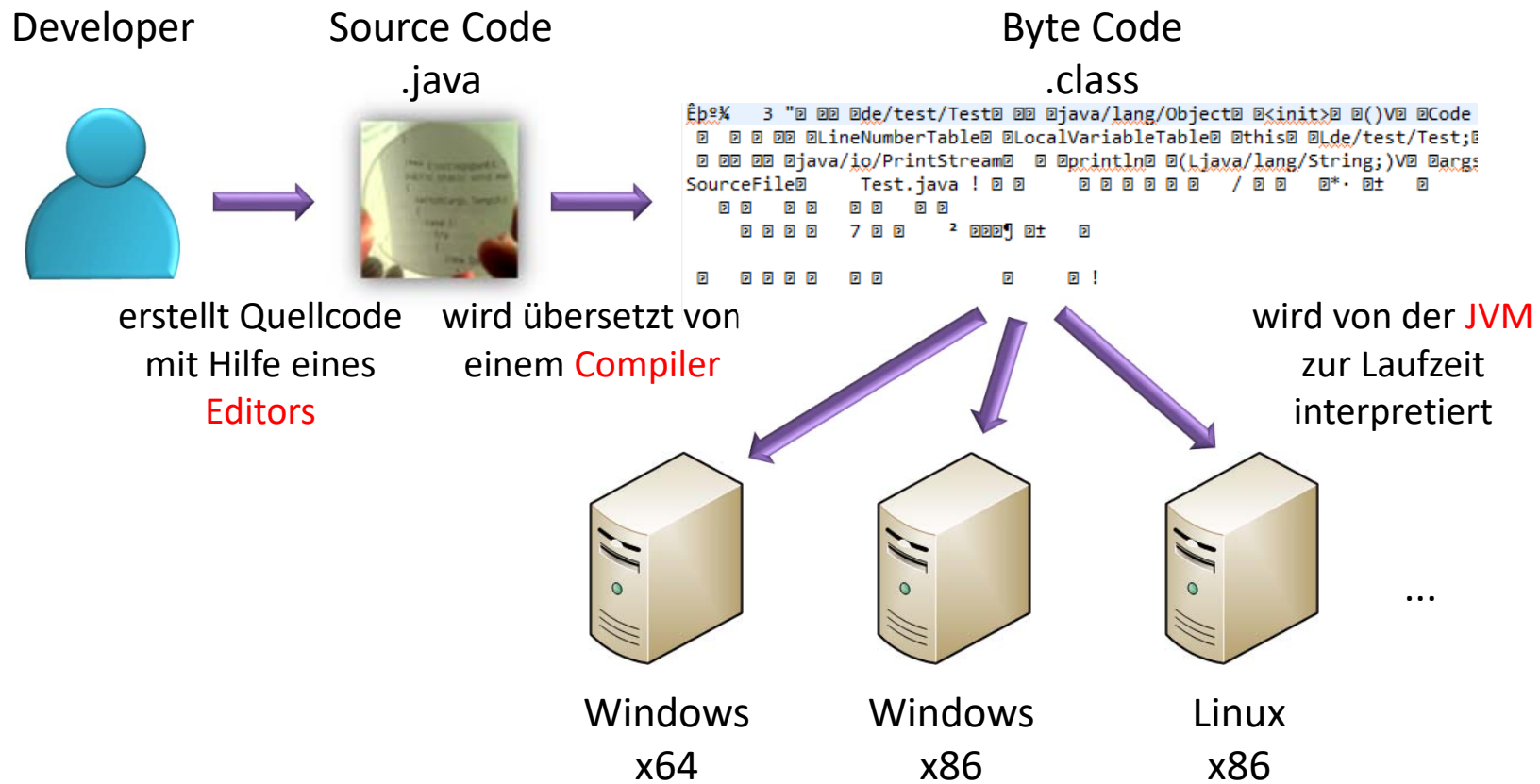
Das erste Java-Programm

Vom Quellcode bis zur Ausführung

Erstellen des Quellcodes

Übersetzen in Bytecode

Ausführen des ersten Programms



Eine Klasse kann mit folgendem Befehl ausgeführt werden, wenn Sie eine main-Methode enthält:

```
java <classname>
```

In unserem Beispiel:

```
java Application
```

Hinweis:

- Die Datei heißt *Application.class*
- Zum Starten wird nur der Klassenname *Application* angegeben.

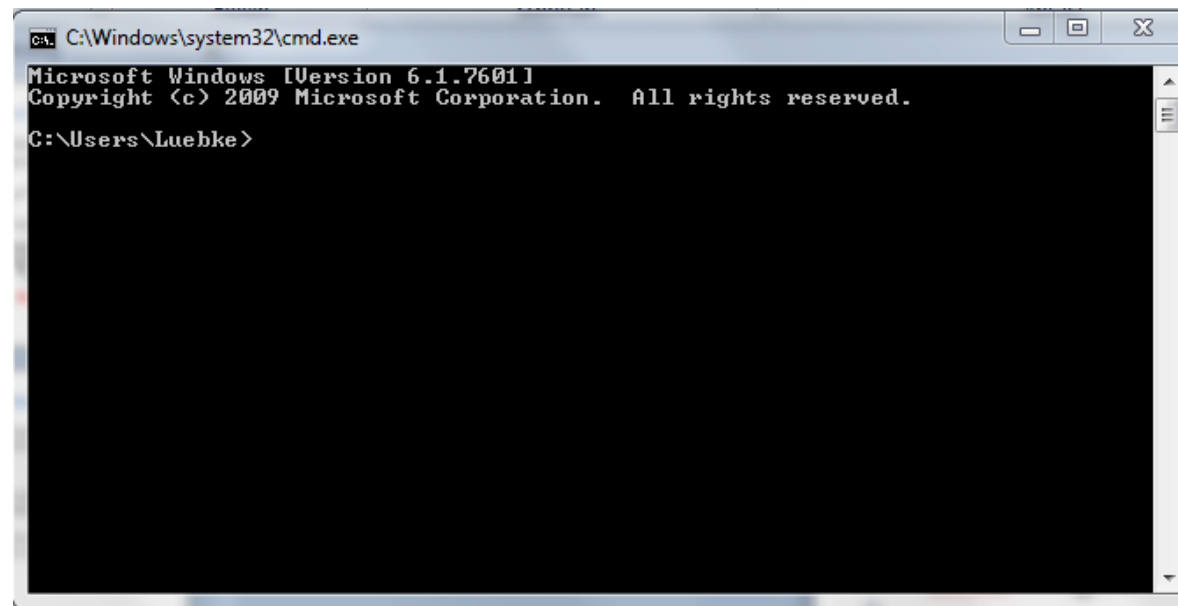
Exkurs: Windows Betriebssystem

Exkurs: Windows Betriebssystem

- Starten Sie den Editor, indem sie
 - unter Start -> Ausführen
 - oder im Feld Start -> Search Program Files*notepad* eingeben.
- Erstellen Sie die Datei Application.java mit ihrem Programmcode.

Exkurs: Windows Betriebssystem

- Starten Sie die Eingabeaufforderung/command line, indem sie
 - unter Start -> Ausführen
 - oder im Feld Start -> Search Program Files*cmd* eingeben.
- Ein Konsolenfenster sollte sich daraufhin öffnen



Exkurs: Windows Betriebssystem

- Im obigen Beispiel ist C:\Users\Luebke der Ordner, in dem sie sich gerade befinden. Sie können von dort aus an die Stelle navigieren, an der Sie die Datei Application.java erstellt haben.
- Zunächst rufen sie mit dem Befehl *dir* den Inhalt des aktuellen Verzeichnisses ab:

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Luebke>dir
Volume in drive C has no label.
Volume Serial Number is 70AB-1BE9

Directory of C:\Users\Luebke

14.10.2012  17:33    <DIR>          .
14.10.2012  17:33    <DIR>          ..
26.03.2012  14:11    <DIR>          .android
13.07.2012  08:19    <DIR>          Contacts
16.10.2012  10:02    <DIR>          Desktop
09.10.2012  22:11    <DIR>          Documents
14.10.2012  22:04    <DIR>          Downloads
13.07.2012  08:19    <DIR>          Favorites
14.10.2012  22:23    <DIR>          Links
13.07.2012  08:19    <DIR>          Music
27.09.2012  13:09    <DIR>          Pictures
13.07.2012  08:19    <DIR>          Saved Games
14.10.2012  22:27    <DIR>          Searches
13.07.2012  08:19    <DIR>          Videos
               0 File(s)              0 bytes
               14 Dir(s) 22.424.707.072 bytes free

C:\Users\Luebke>
```


Exkurs: Windows Betriebssystem

- In der Auflistung sehen Sie die verschiedenen Ordner, zu denen Sie direkt navigieren können. Wenn sich die Datei `Application.java` bspw. auf dem Desktop befindet, können sie mit dem `cd`-Befehl (`cd` = change directory) in das Unterverzeichnis (den Ordner) wechseln:

```
C:\Users\Luebke>cd Desktop
```

```
C:\Users\Luebke\Desktop>
```

- Hier können Sie jetzt mit dem Befehl

```
javac Application.java
```

die Datei `Application.java` übersetzen und mit

```
java Application
```

die Datei `Application.class` ausgeführt werden.

- Schauen Sie nach jedem abgesetzten Befehl mit dem Befehl *dir* nach, ob das gewünschte Ergebnis eingetreten ist.

Exkurs: Weitere Befehle zum Ausprobieren

- Mit dem Befehl

```
cd ..
```

wechseln Sie in das darüber liegende Verzeichnis.

- Der Befehl

```
rename Application.java.txt Application.java
```

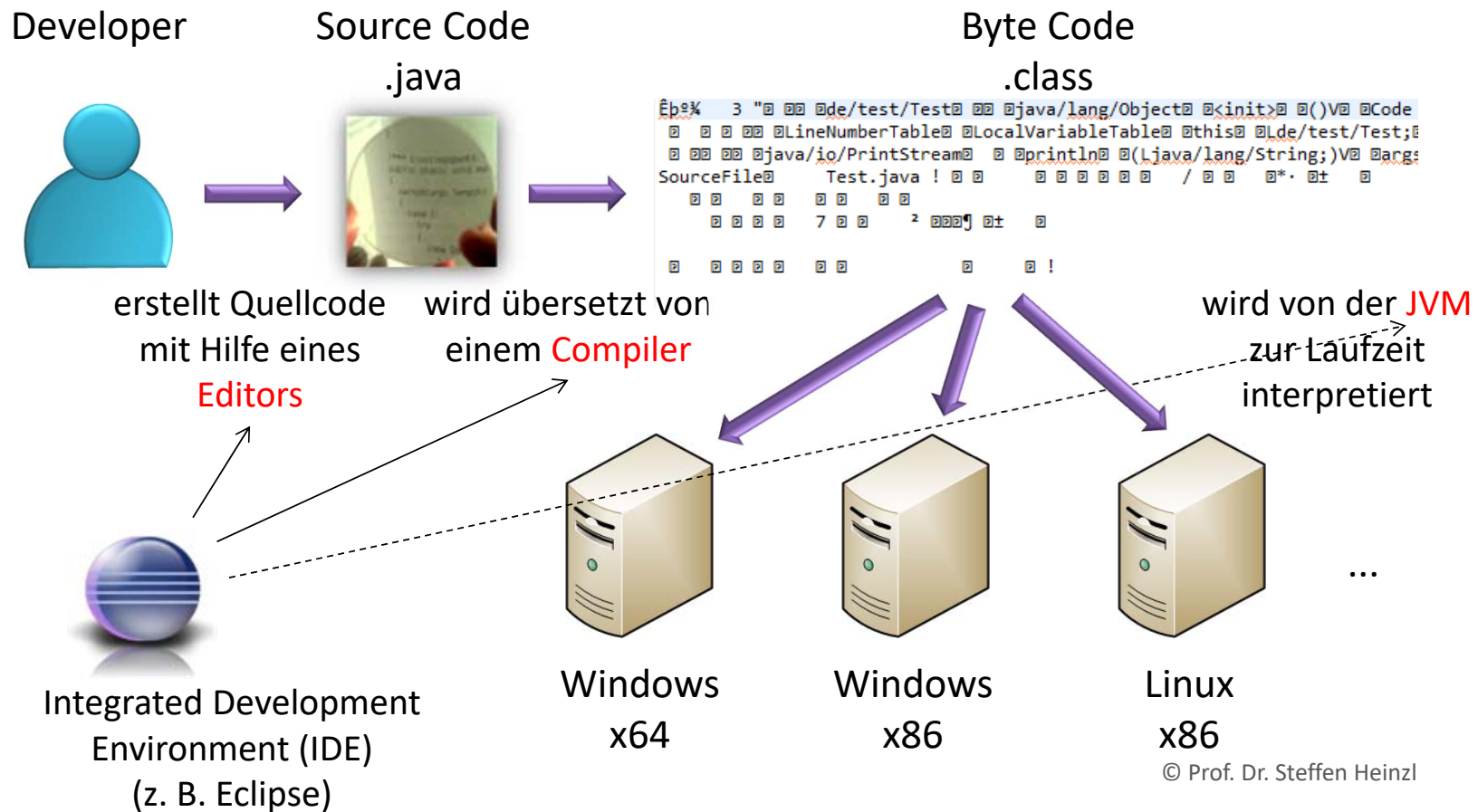
benennt die Datei Application.java.txt in Application.java um.

- Pendants unter Unix und Mac:

- ls entspricht dir
- mv entspricht rename

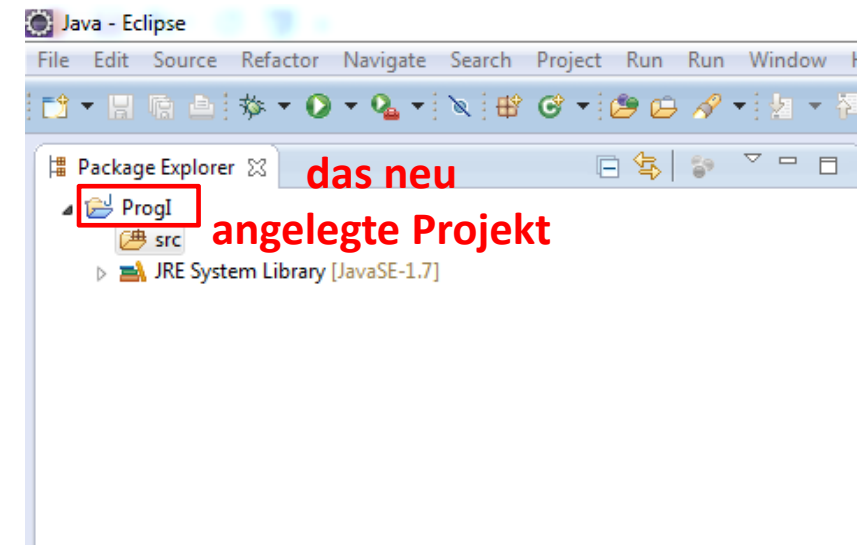
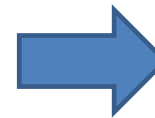
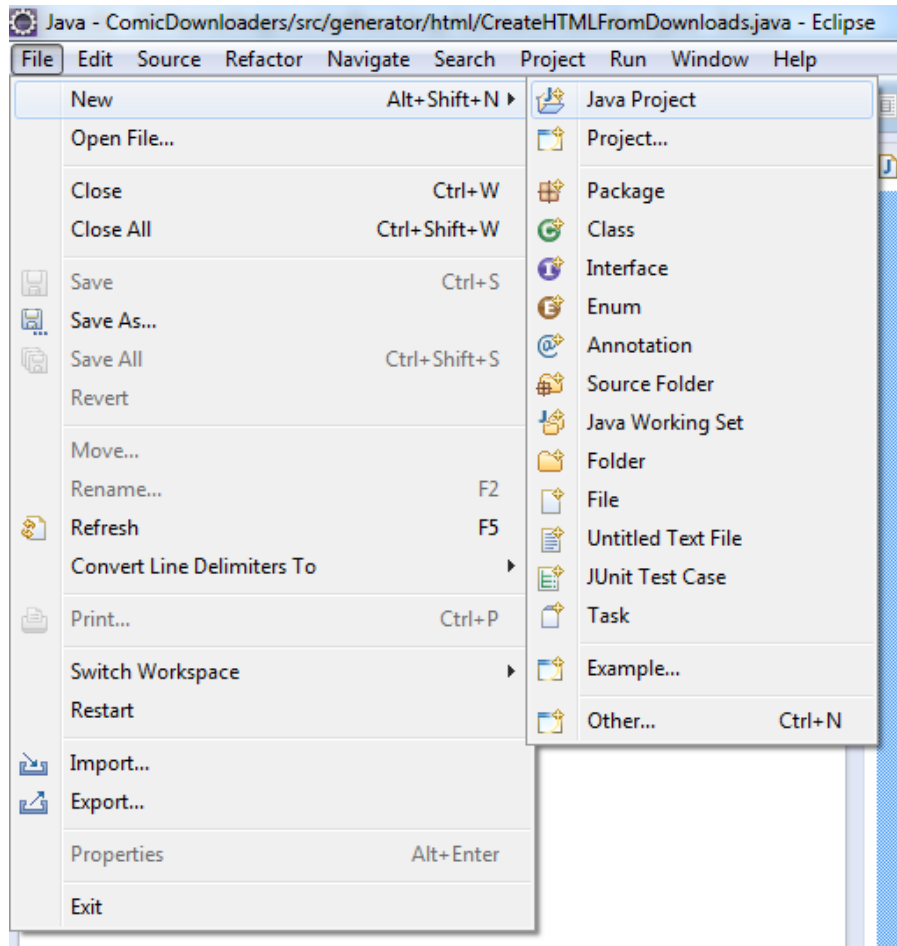
Ende
Exkurs: Windows Betriebssystem

Vom Quellcode bis zur Ausführung

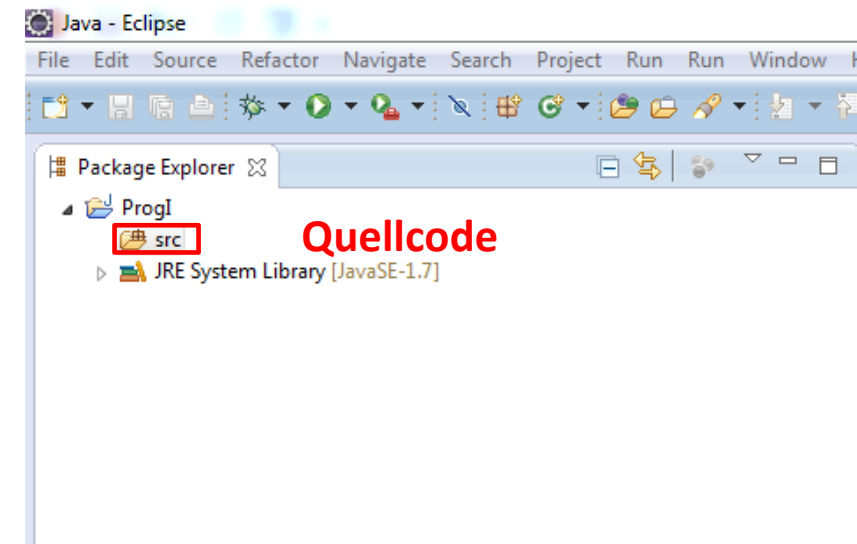
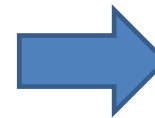
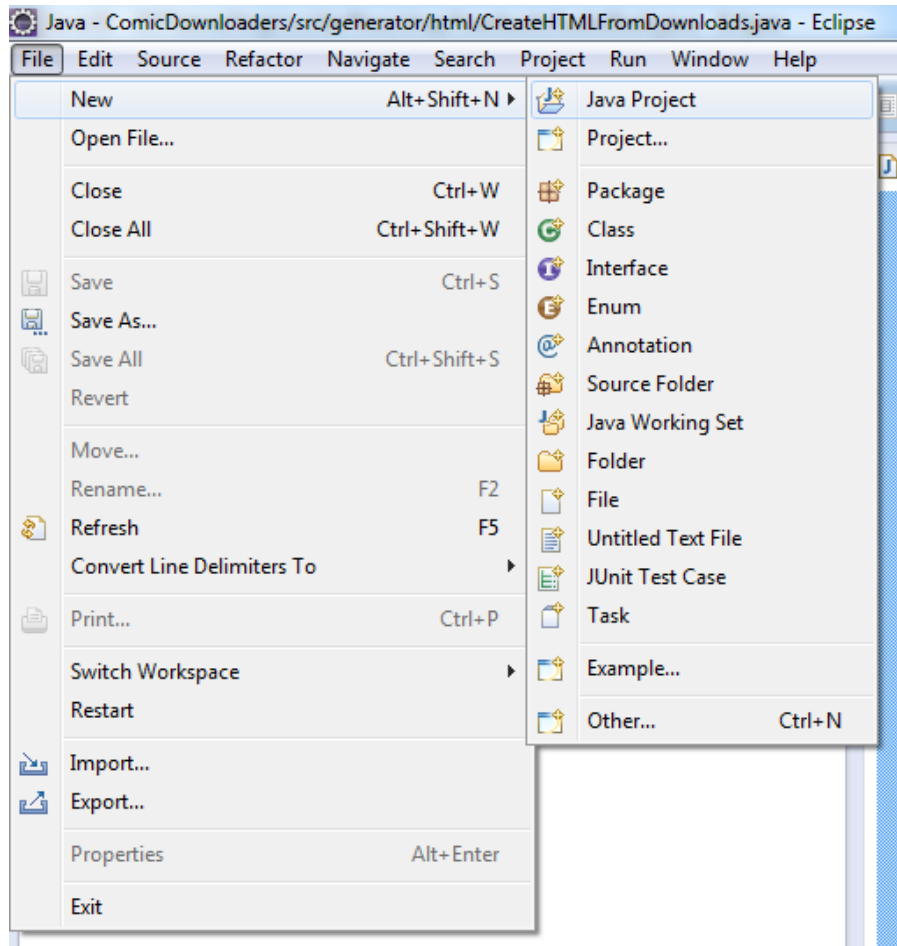


Erstes Java-Programm in Eclipse

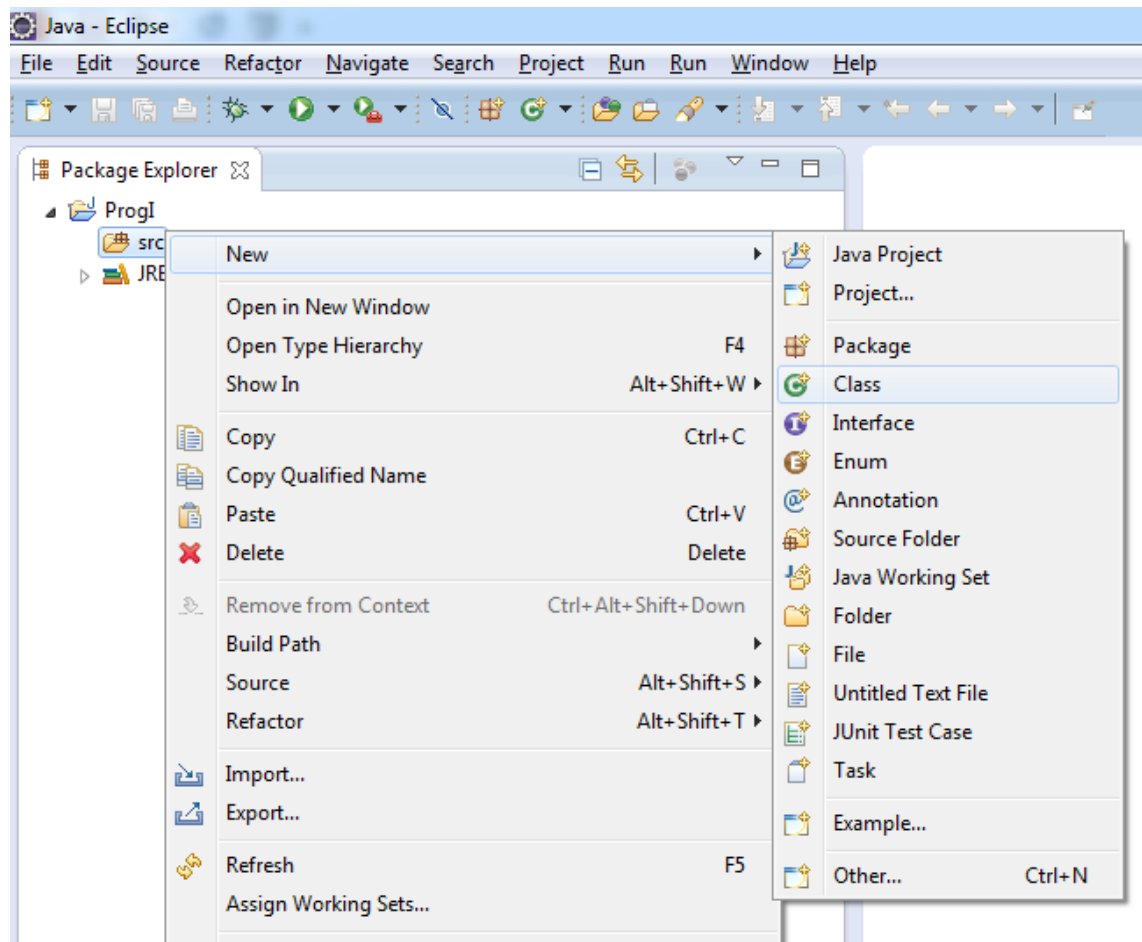
Exkurs Eclipse: Neues Projekt anlegen



Exkurs Eclipse: Neues Projekt anlegen

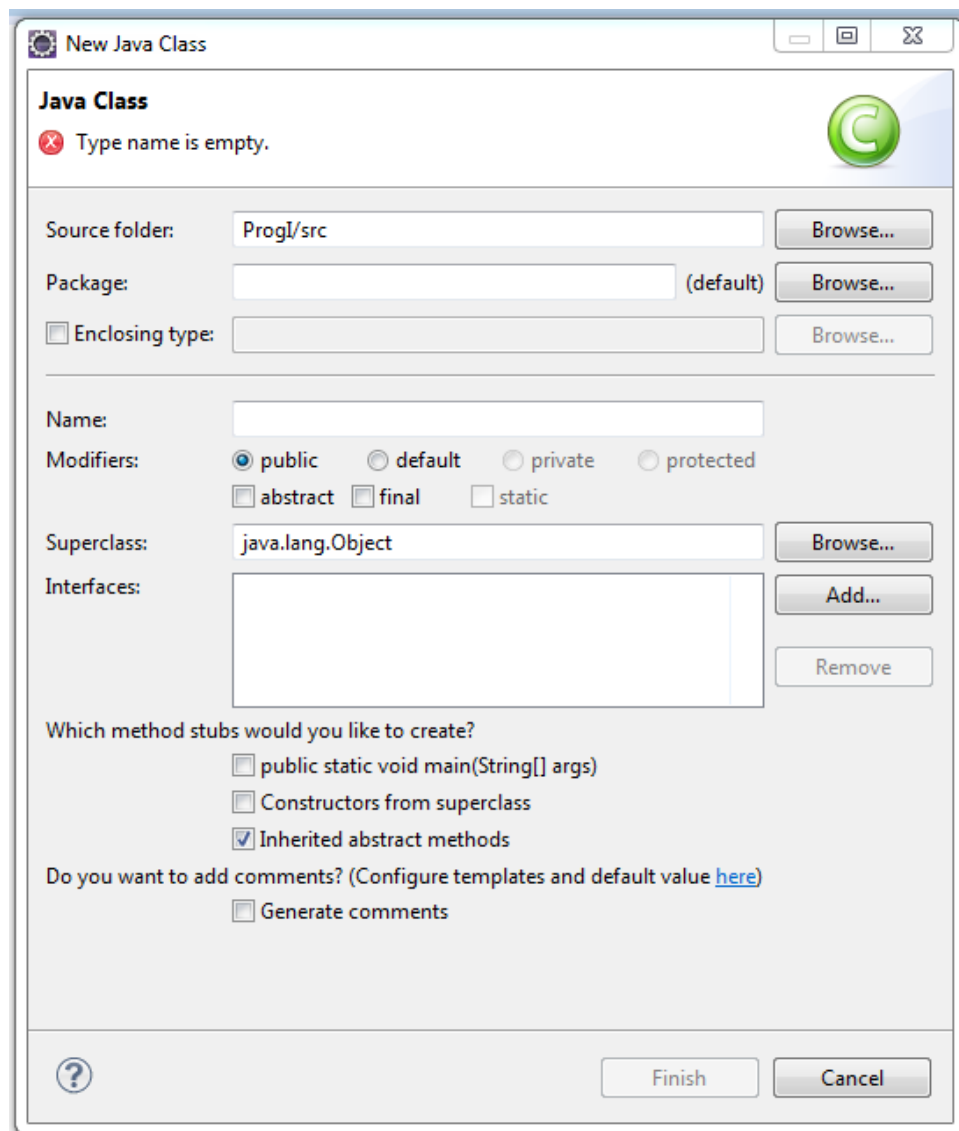


Exkurs Eclipse: Neue Klasse anlegen

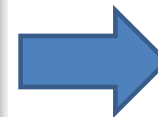


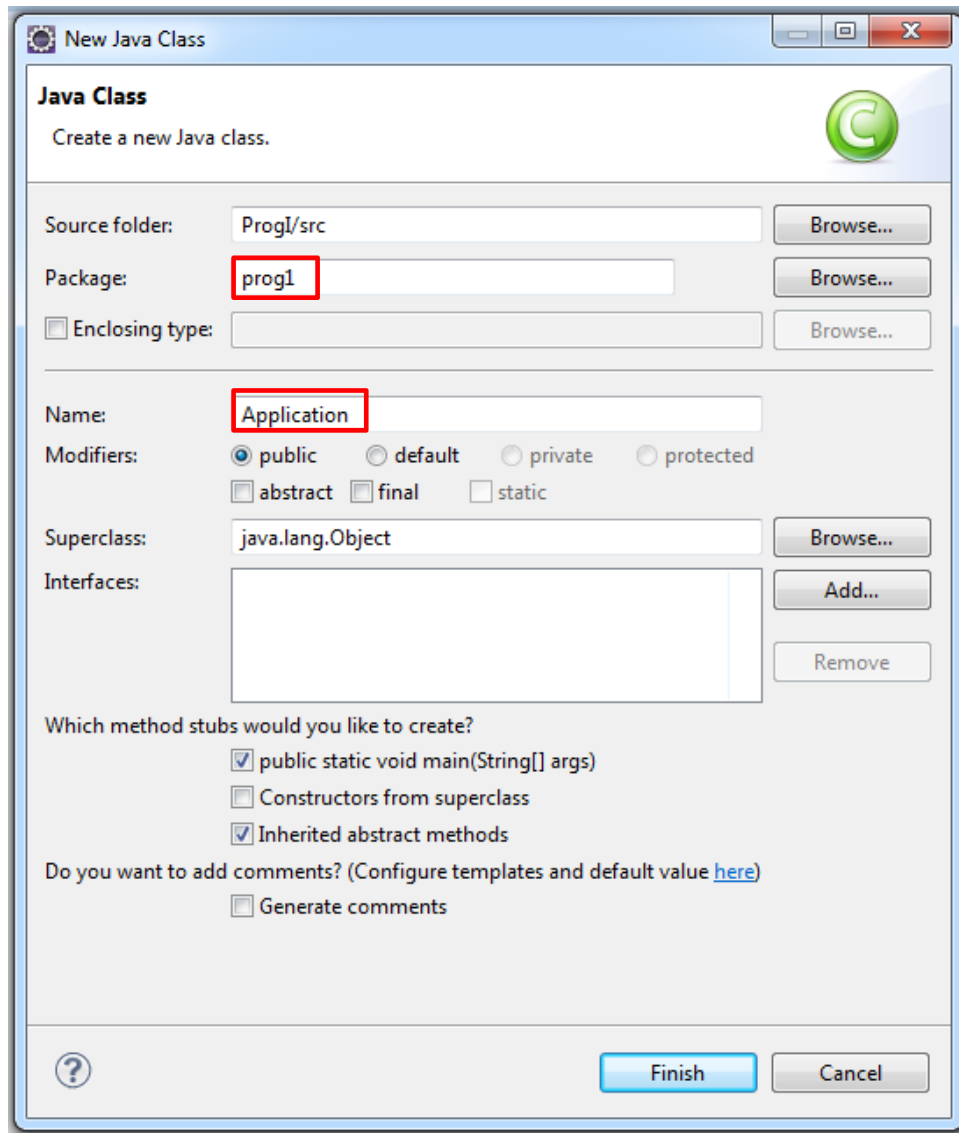
Rechtsklick auf „src“
-> New -> Class

New Java Class Wizard
öffnet sich



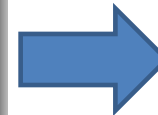
Exkurs Eclipse: New Java Class Wizard





Exkurs Eclipse: New Java Class Wizard ctd.

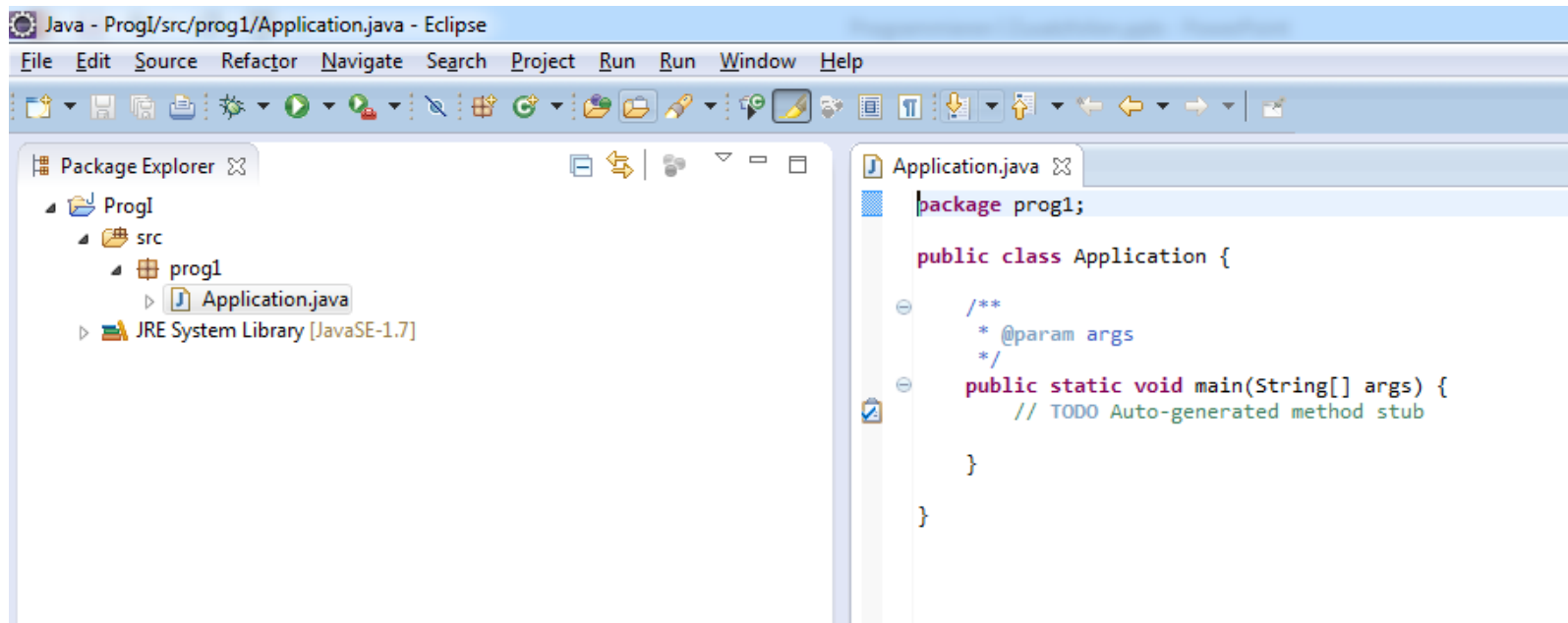
Packages dienen als
Strukturierungsmerkmal bei
größeren/mehreren Programmen



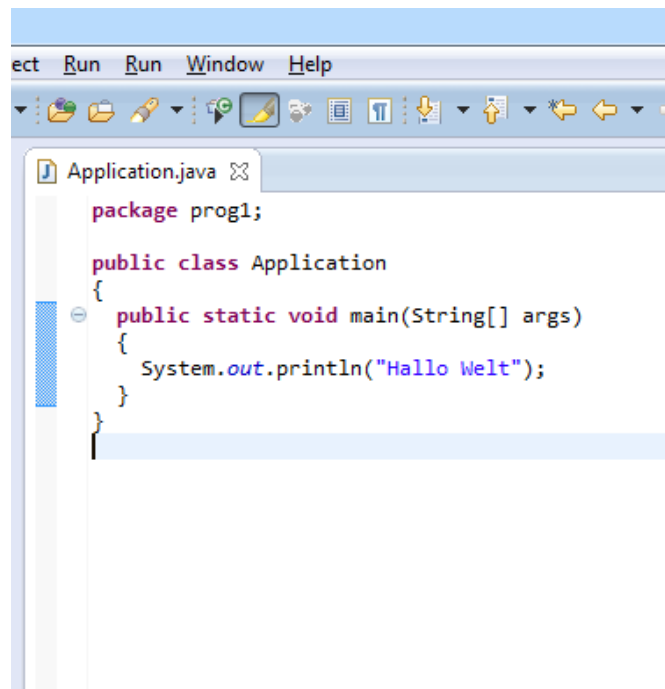
Exkurs Eclipse

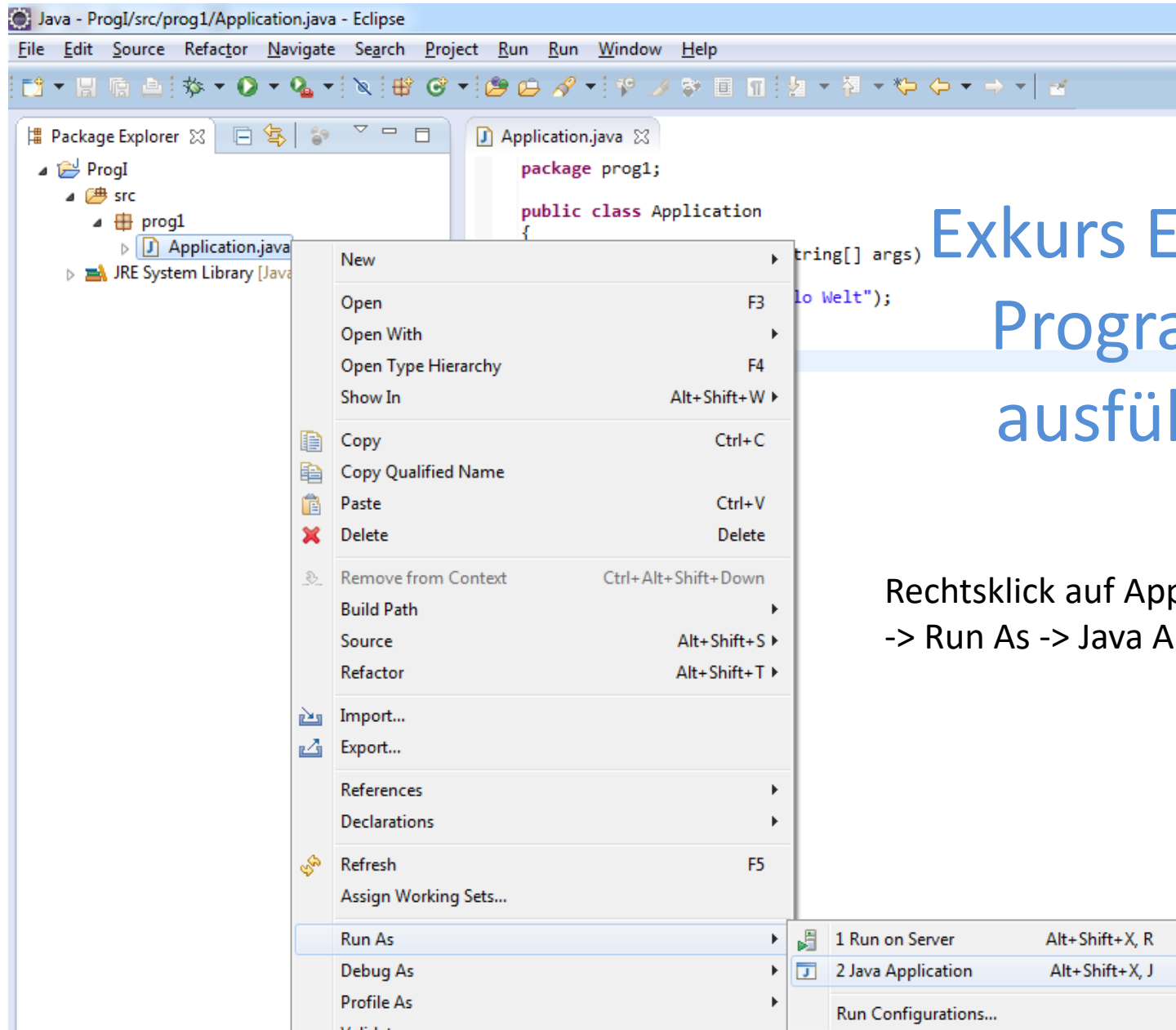
Projektübersicht

Editor



Exkurs Eclipse: Quellcode editieren

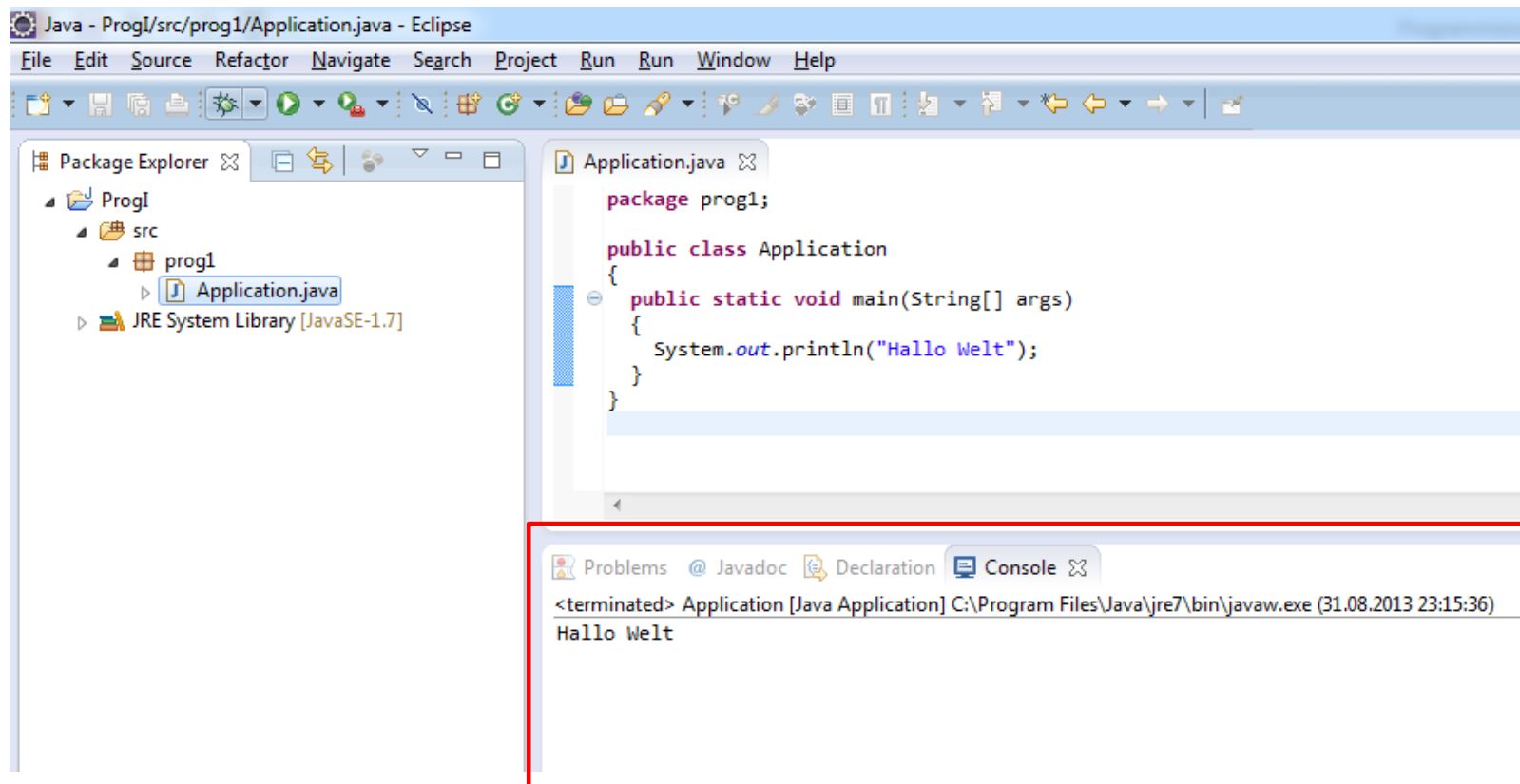




Exkurs Eclipse: Programm ausführen

Rechtsklick auf Application.java
-> Run As -> Java Application

Exkurs Eclipse: Konsolenansicht



Kommentare

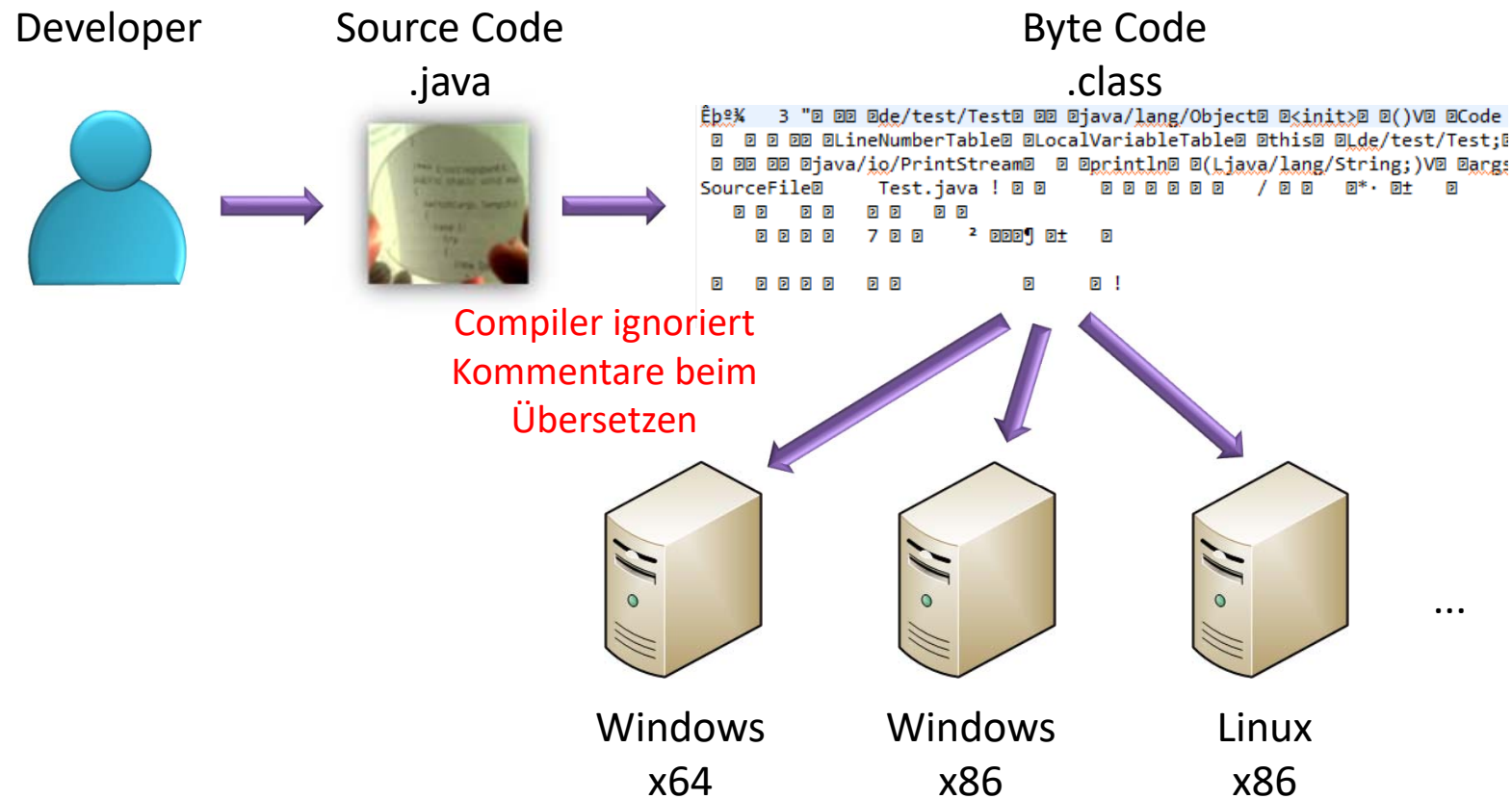
Kommentare – Beispiel

- Anmerkungen zu Befehlen, Klassen, etc. können im Programmcode platziert werden:

```
//Der Name der Klasse ist Application
public class Application
{
    /*Der Einstiegspunkt in ein Programm
    ist die main-Methode */
    public static void main(String[] args) {
        //Gibt den Text Hallo Welt auf dem Bildschirm aus
        System.out.println("Hallo Welt");
    }
}
```

**Zeichenkette bzw.
String**

Kommentare werden beim Übersetzen ignoriert



Kommentare

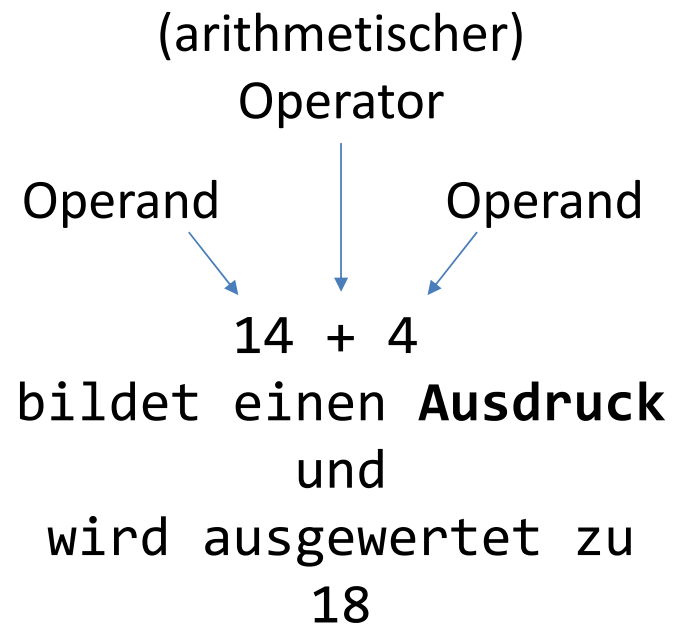
- Kommentare werden vom Compiler ignoriert und dienen dazu dem Quellcode Anmerkungen bzw. Erklärungen hinzuzufügen

- Java kennt drei Arten von Kommentar:

- Blockkommentar `/* ... */`
- Zeilenkommentar `//`
- Dokumentationskommentar (Java-Doc) `/** ... */`

Ausdrücke und Operatoren

Ausdruck



Rechnen in Java

Arithmetische Operatoren

In der Mathematik:

- Addition

$$14 + 4 = 18$$

- Subtraktion

$$14 - 4 = 10$$

- Multiplikation

$$14 \cdot 4 = 56$$

$$4a$$

In Java:

- Addition

`14 + 4` wird ausgewertet zu `18`

- Subtraktion

`14 - 4` wird ausgewertet zu `10`

- Multiplikation

`14 * 4` wird ausgewertet zu `56`

$$4 * a$$

Rechnen in Java

Arithmetische Operatoren

In der Mathematik:

- Ganzzahlige Division

$14 : 4 = 3 \text{ Rest } 2$

- Modulo

$14 \bmod 4 = 2$

- Reellwertige Division

$14 : 4 = 3,5$

$24,75 : 4,5 = 5,5$

In Java:

- Ganzzahlige Division

$14 / 4$ wird ausgewertet zu **3**

- Modulo

$14 \% 4$ wird ausgewertet zu **2**

- Reellwertige Division

$14.0 / 4.0$ wird ausgewertet zu **3.5**

$24.75 / 4.5$ wird ausgewertet zu **5.5**

Rechnen in Java

Modulo Operator

$a \% b$
entspricht
 $a - (a / b) * b$



$14 \% 4$
entspricht
 $14 - (14 / 4) * 4$

Rechnen in Java

Übersicht: Arithmetische Operatoren

$$\begin{array}{c} 14 + 4 \\ \hline 18 \end{array}$$

Addition

$$\begin{array}{c} 14 - 4 \\ \hline 10 \end{array}$$

Subtraktion

$$\begin{array}{c} 14 * 4 \\ \hline 56 \end{array}$$

Multiplikation

$$\begin{array}{c} 14 / 4 \\ \hline 3 \end{array}$$

ganzzahlige(!)
Division

$$\begin{array}{c} 14 \% 4 \\ \hline 2 \end{array}$$

Modulo

$$\begin{array}{c} 14.0 / 4.0 \\ \hline 3.5 \end{array}$$

reellwertige
Division

Auswertung von Ausdrücken

- Wie in der Mathematik gilt Punkt-vor-Strich-Rechnung sowie Klammersetzung:

$$\begin{array}{ccccccc} 2 & * & 3 & + & (8 & - & 5) & / & 6 \\ & & & & \underbrace{} & & & & \\ 2 & * & 3 & + & 3 & & & / & 6 \\ \underbrace{} & & & & & & & & \\ 6 & & + & & 3 & & & / & 6 \\ & & & & \underbrace{} & & & & \\ 6 & & + & & & & 0 & & \\ \underbrace{} & & & & & & & & \\ & & & & 6 & & & & \end{array}$$

- Was macht man mit den Ergebnissen?
 - Ausgeben/Zwischenspeichern

Auswertung von Ausdrücken

- Die Auswertung von Ausdrücken erfolgt in folgender Reihenfolge

Operator(en)	Priorität
[] . (<parameters>) expr++ expr--	1 (höchste)
++expr --expr +expr -expr ~ ! (<type>) new	2
* / %	3
+ -	4
<< >> >>>	5
< <= > >= instanceof	6
== !=	7
&	8
^	9
	10
&&	11
	12
? :	13
= += -= *= /= %= <<= >>= >>>= &= ^= =	14 (niedrigste)

Einfache Ausgabe

Wir können jetzt Berechnungen anstellen,
aber wie geben wir das Ergebnis aus?

Java stellt eine **Methode** zur Verfügung,
um Text auf der Standardausgabe auszugeben.

Ausgabe

- Die Ausgabe auf den Bildschirm erfolgt über `System.out.println`
- `System.out.println` nimmt ein **Argument** entgegen

```
public class Application {  
    public static void main(String[] args) {  
        /*Argument wird ausgewertet und dann ausgegeben*/  
        System.out.println(14 * 4);  
    }  
}
```

`println` ist eine Methode

Der Teil innerhalb der Klammern wird an die Methode übergeben und **Argument** genannt.

Variablen und Zuweisungen

Komplexere Berechnungen können schnell unübersichtlich werden.

Beispiel: Summe der ersten n Quadratzahlen

$$\frac{1}{6}n(n+1)(2n+1)$$

Für n = 5:

```
public class SummeNQuadratZahlen1
{
    public static void main(String[] args)
    {
        System.out.println((1.0/6.0)*5*(5+1)*(2*5+1));
    }
}
```

➤ Eine Möglichkeit zur Zwischenspeicherung hilft.

Wenn wir die Summe der ersten 8 Quadratzahlen berechnen wollen, müssen wir die Berechnung neu machen.

Für $n = 8$:

```
public class SummeNQuadratZahlen2
{
    public static void main(String[] args)
    {
        System.out.println((1.0/6.0)*8*(8+1)*(2*8+1));
    }
}
```

Wäre es nicht besser, den Wert an nur einer Stelle von 5 auf 8 ändern zu müssen?

➤ Eine Möglichkeit zur Zwischenspeicherung hilft.


```
public class SummeNQuadratZahlen3
{
    public static void main(String[] args)
    {
        int n = 8;
        System.out.println((1.0/6.0)*n*(n+1)*(2*n+1));
    }
}
```

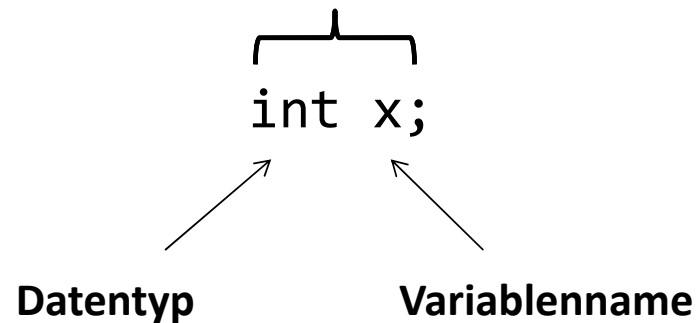
Wir speichern die 8 in einer Variablen zwischen.

Variablen

- Werte können in Variablen zwischengespeichert werden.
- Eine **Variable** dient als benannter Speicherplatz.
- `int x;` reserviert einen Speicherplatz für eine ganze Zahl (Integer).

Die Variable x wird deklariert.

Variablendeklaration



Konventionen für Variablennamen

- Variablen sollen **sprechende Namen** haben.
- Variablennamen beginnen immer mit einem Kleinbuchstaben.

```
int counter;
```

- zusammengesetzte Variablennamen „verwenden“ **Camel Case**

```
int numberOfTracks;
```



Jedes weitere Wort beginnt mit einem Großbuchstaben

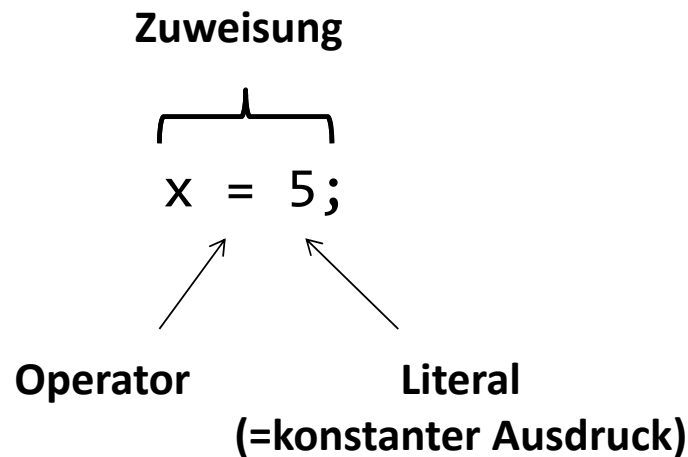
- Variablennamen dürfen nicht mit Zahlen beginnen

```
int 2a;
```

- Generell sollten Namen (Variablen-, Methoden-, Klassen-, Package-, Projektnamen) niemals Umlaute enthalten

Zuweisung

- Der Zugriff auf den Speicherplatz erfolgt über den Variablennamen x.
- Die Anweisung `x = 5;` hinterlegt in dem Speicherplatz x den Wert 5.
- Man spricht auch von einem schreibenden Zugriff.

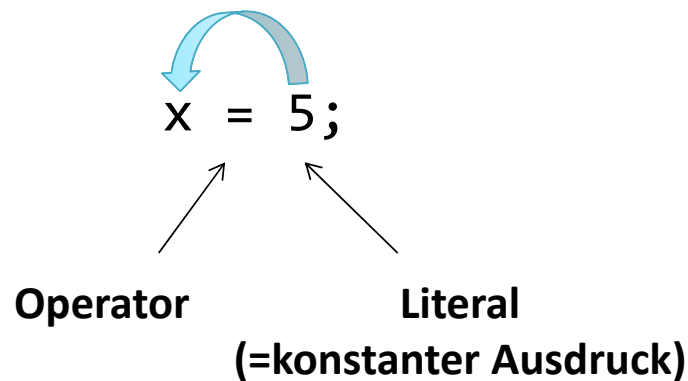


Zuweisung

- Der Zugriff auf den Speicherplatz erfolgt über den Variablennamen x.
- Die Anweisung `x = 5;` hinterlegt in dem Speicherplatz x den Wert 5.
- Man spricht auch von einem schreibenden Zugriff.

**Der Variablen x wird der Wert 5 zugewiesen.
Die Zuweisung erfolgt von rechts nach links.**

Zuweisung



Übersicht Variablen

```
/*reserviert einen Speicherplatz für eine ganze Zahl.*/
```

```
int x;
```

```
/*schreibt die Zahl 5 in die Variable x, d.h. in den Speicherplatz, der über  
den Variablennamen x angesprochen wird.*/
```

```
x = 5;
```

```
/*liest den Wert aus der Variablen x, d.h. von dem Speicherplatz, der durch x  
repräsentiert wird, und gibt diesen aus*/
```

```
System.out.println(x);
```

```
/*reserviert einen Speicherplatz für eine ganze Zahl und schreibt in diesen  
Speicherplatz direkt eine 3.*/
```

```
int y = 3;
```


Variablen – Beispiel

```
public class Application {  
    public static void main(String[] args) {  
        /*Variablendeklaration: Deklariert die Variable ergebnis  
        vom Typ int (Integer, d.h. Ganzzahl)*/  
        int ergebnis;  
  
        /*Definiert die Variable x. Unter Definition versteht man  
        die Deklaration inkl. Initialisierung (d.h. erste  
        Wertbelegung) einer Variablen*/  
        int x = 5;  
        int y = 7; //Definiert die Variable y.  
        ergebnis = x * y; //Multipliziere die Variable x mit y  
        /*Ergebnis auf Standardausgabe ausgeben*/  
        System.out.println(ergebnis);  
    }  
}
```

Variablen – Beispiel

- Das gleiche Programm noch mal ohne Kommentare:

```
public class Application {  
    public static void main(String[] args) {  
        int ergebnis;  
        int x = 5;  
        int y = 7;  
        ergebnis = x * y;  
        System.out.println(ergebnis);  
    }  
}
```



Anweisungen
werden
sequentiell
ausgeführt