



LUDWIG-
MAXIMILIANS-
UNIVERSITÄT
MÜNCHEN



 mobile and
distributed systems group



Javakurs für Anfänger

Einheit 01: Organisation & Einführung

Lorenz Schauer

Lehrstuhl für Mobile und Verteilte Systeme



Organisatorisches

- Ziele und Aufbau
- Zielgruppe, Vergütung, Webseite
- Kontakt

Einführung in Java

- Motivation
- Compiler & Interpreter
- Geschichte
- Eigenschaften von Java

Zum ersten Programm

- Installation
- Der Weg zum ersten Programm
- Hallo Welt!
- IDEs (Entwicklungsumgebung)

Lernziele

- Motivation zur Programmierung mit Java
- Installation der notwendigen Programme
- Erstes Java-Programm schreiben und ausführen können

Organisatorisches zum Kurs

- **Ziele:**
 - Selbstständiges Programmieren lernen mit Java
 - Keine Vorkenntnisse notwendig!
 - Kompakte und v.a. praktische Einarbeitung in die Grundlagen der objektorientierten Programmierung mit Java
 - Vermittlung von notwendigen theoretische Grundlagen und Konzepte der Programmierung mit Java

- **Aufbau:**
 - Praktische Ergänzung zur Vorlesung „Einführung in die Programmierung“
 - Mischung aus Vorlesung und praktischen Programmierereinheiten
 - Kleinere Programmieraufgaben müssen während der Veranstaltung selbstständig gelöst werden (ggf. mit Hilfestellung)
 - Bitte bringen Sie daher auch immer Ihr eigenes Gerät (Laptop) mit!

- **Zielgruppe:**
 - V.a. Bachelorstudenten mit Nebenfach Informatik
 - Studenten mit Hauptfach Informatik zur praktischen Verbesserung ihrer Programmierkenntnisse in Java

- **Vergütung:**
 - Bachelorstudenten mit Nebenfach Informatik (zu 60 ECTS) erhalten für die erfolgreiche Teilnahme am Kurs 3 ECTS
 - **Dürfen max. 2 Mal fehlen -> Anwesenheitskontrolle!**

- **Ort und Zeit:**
 - Ab 20.10.2016 immer Donnerstags, von 18.00 -20.00 Uhr c.t.
 - Oettingenstraße 67, Raum: B 001

- **Webseite:**
 - <http://www.mobile.ifi.lmu.de/lehveranstaltungen/java-fuer-anfaenger-ws1617/>
 - Obligatorische Anmeldung zum Kurs über Uniworx (<https://uniworx.ifi.lmu.de/>)

▪ Veranstalter:

- Lorenz Schauer (Wiss. Mitarbeiter)
 - Büro:
 - Lehrstuhl für Mobile und Verteilte Systeme
Oettingenstraße 67, Raum U160
 - Sprechstunde:
 - Montags, 10 - 12 Uhr
 - Donnerstags, 14.00 - 16.00 Uhr
 - Kontakt:
 - Mail: lorenz.schauer@ifi.lmu.de
 - Tel.: 089-2180-9157
 - Web: <http://www.mobile.ifi.lmu.de/team/lorenz-schauer/>

- Tutor:
 - Daniel Bauersachs: D.Bauersachs@campus.lmu.de



Einführung (Heute)

- IDE und erstes Programm

Klassen & Objekte

- Einführung in die Objektorientierung

Erweiterter Programmablauf & Kontrollstrukturen

- If, then, else, ...

Felder (Arrays)

- Arbeiten mit statischen und dynamischen Arrays

Mehr zur Objektorientierung

- Klassenvariablen u. Klassenmethoden, Vererbung, Abstrakte Klassen
- Schnittstellen (Interfaces)

Umgang mit UML und Java API

Ausnahmen (Exceptions)

Generische Klassen (Generics)



Matthias Hölzl, Allaithy Raed, Martin Wirsing:
Java Kompakt: Eine Einführung in die Software-Entwicklung mit Java,
Springer-Verlag.

Für Studenten (PAC-Proxy)¹ kostenlos online auf:

<http://link.springer.com/book/10.1007/978-3-642-28504-2>



Christian Ullenboom:

Java ist auch eine Insel,

Rheinwerk Computing, ISBN 978-3-8362-1802-3.

Online (kostenlos): <http://openbook.rheinwerk-verlag.de/javainsel/>



Guido Krüger, Heiko Hansen:

Java-Programmierung Das Handbuch zu Java 8,

Die HTML-Ausgabe der 7. Auflage (Stand 2011) kann kostenlos

heruntergeladen werden: <http://javabuch.de/download.html>

1: Hinweise zum LRZ PAC-Proxy auf <https://www.lrz.de/services/netzdienste/proxy/zeitschriftenzugang/>



Hans-Peter Habelitz:
 Programmieren lernen mit Java, Rheinwerk Verlag GmbH
 3. Auflage
 ISBN: 978-3-8362-3517-4, ca. 19,90 Euro



LUDWIG-
MAXIMILIANS-
UNIVERSITÄT
MÜNCHEN



 mobile and
distributed systems group



Teil 1: Motivation

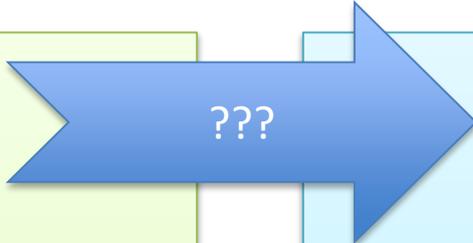
Warum Java?



Grundproblem:

Menschen besitzen komplexe Sprache (Subjekt – Prädikat – Objekt). Das Verstehen von Maschinencode ist umständlich

```
0011 1100 0001 1101 1100 1000
0110 0011 1010 1011 1001 1100
0010 0101 0011 1110 0110 0011
0011 01.....
```



???

Maschinen besitzen eine sehr primitive Sprache (Befehlsfolge) und können nur Maschinensprache verstehen:

```
0011 1100 0001 1101 1100 1000
0110 0011 1010 1011 1001 1100
0010 0101 0011 1110 0110 0011
0011 01.....
```

Lösung:

- Menschen nutzen einen anderen Befehlssatz als Sprache L1 (Bsp.: C++, Java, Python, usw.)
- Maschinen verarbeiten eine Übersetzung (**Compiler**) bzw. Interpretation (**Interpreter**) von L1, hier als L0 bezeichnet.

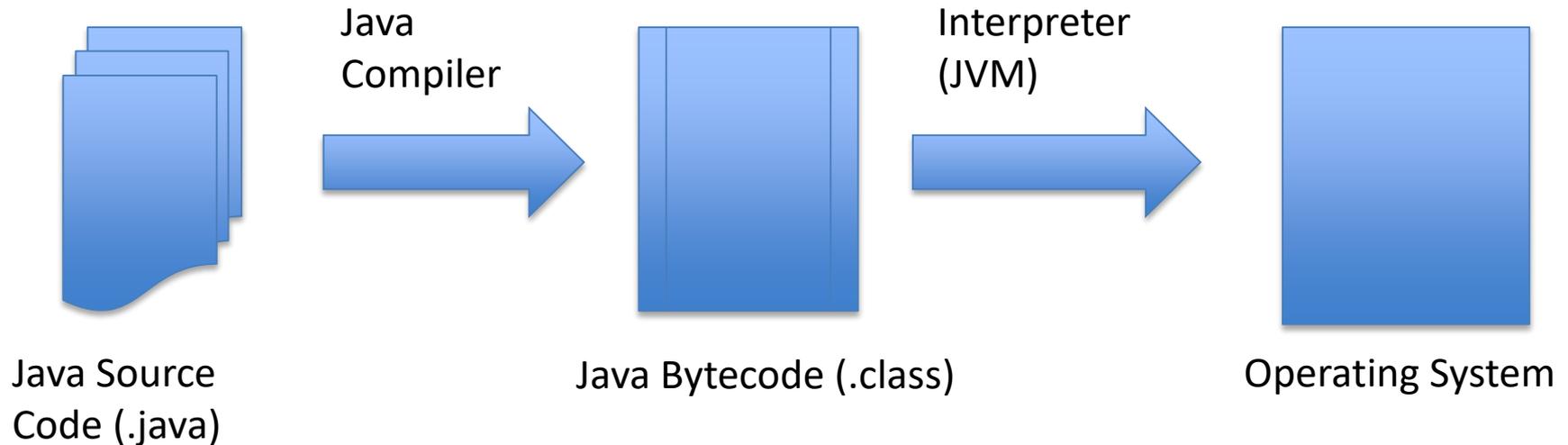
▪ **Compiler:**

- Vollständige Übersetzung des Programms von L1 zu L0
- Quellprogramm in L1 wird verworfen
- Zielprogramm in L0 wird in Speicher geladen und ausgeführt

▪ **Interpreter:**

- Jede L1 Anweisung wird analysiert, dekodiert und unmittelbar in L0 ausgeführt
- Quellcode + Interpreter auf ausführender Maschine benötigt
- I.d.R. langsamer als kompilierte Programme

Auch hybride Ansätze möglich: Beispiel Java

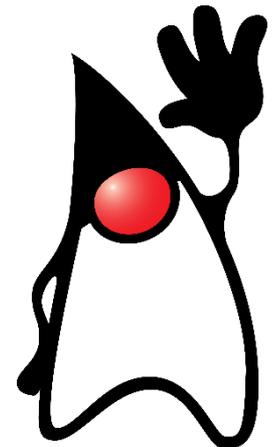


- Quellcode wird von Java Compiler in Bytecode übersetzt
- JVM interpretiert den Bytestream als nativen Maschinencode, der vom Betriebssystem ausgeführt werden kann

Was hat eine indonesische Insel mit Java als Programmiersprache zu tun?

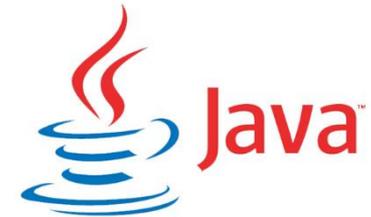
Geschichtlicher Überblick:

- 1991 bei *Sun Microsystems* als Software für einen programmierbaren Prototypen entworfen
 - Ziele: Klein, effizient, stabil und sicher
 - Objektorientierte Sprache mit Anlehnung an C++ Syntax
 - Urspr. Oak (**O**bject **A**pplication **K**ernel) inspiriert durch eine Eiche am Bürofenster
- Geblieben aus dem „Green Project“ nur die Programmiersprache und *Duke*, das heutige Java Maskottchen
- Programmiersprache nun im Fokus
 - Klein, objektorientiert, plattformunabhängig und robust passend zum aufkommenden WWW
 - Anderer Name musste her, da Oak bereits geschützt:
 - Wahrscheinlich Einigung bei einer Tasse Kaffee (Sorte: Java)



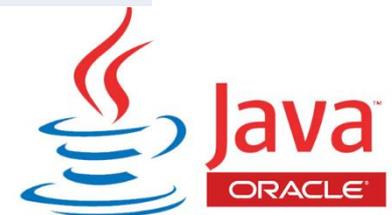
- 23. Mai 1995: Java wird öffentlich vorgestellt
 - Ankündigung: Integration in Netscape Navigator 2.0

- 1996: Veröffentlichung des 1. Java Development Kit (*JDK 1.0*)
 - Seitdem zahlreiche Erweiterungen. Hier ein kleiner Auszug:



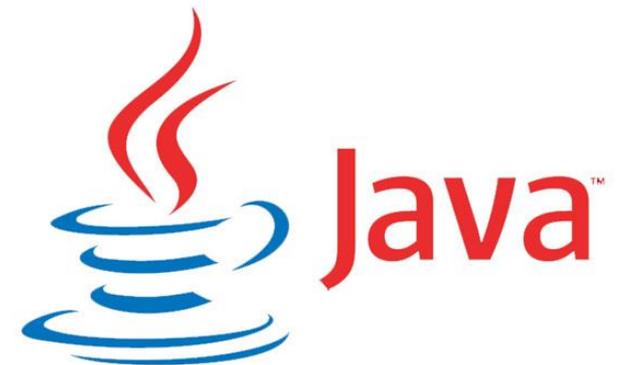
Jahr/Version	#Klassen/Interfaces	Neuheiten
1998/ V 1.2	1520	Just-in-Time Compiler, Swing, ...
2000/ V 1.3	1840	Sound-API, ...
2004/ V 5	3279	Generics, ...
2011/ V 7	4024	Neue Filesystem-API, ...
2014/ V 8	4240	Lamda-Funktionen, ...
Vorr. 2017/V 9	++	Änderungen an JVM gelpant

- Januar 2010: *Oracel* kauft *Sun* und treibt Java-Entwicklung weiter voran



Eigenschaften von Java:

- **Hochsprache** mit komplexerer Syntax und Grammatik (im Vergleich zu maschinennahen Sprachen)
- Einfachheit (im Vergleich zu anderen obj. orientierten Sprachen, wie C++)
- Robustheit (Autom. Speicherverwaltung)
- Kombiniert Vorteile von Compiler und Interpreter
- Universell einsetzbar
 - U.a. verteilte Systeme und Netze, Inet-Anwendungen, usw.
- Plattformunabhängig durch JVM
- Portabilität
- Objektorientiert
- U.v.m.





LUDWIG-
MAXIMILIANS-
UNIVERSITÄT
MÜNCHEN



Teil 2: Zum ersten Programm

Installation und Hello World!



Die 3 wichtigsten Installationsvarianten von Java

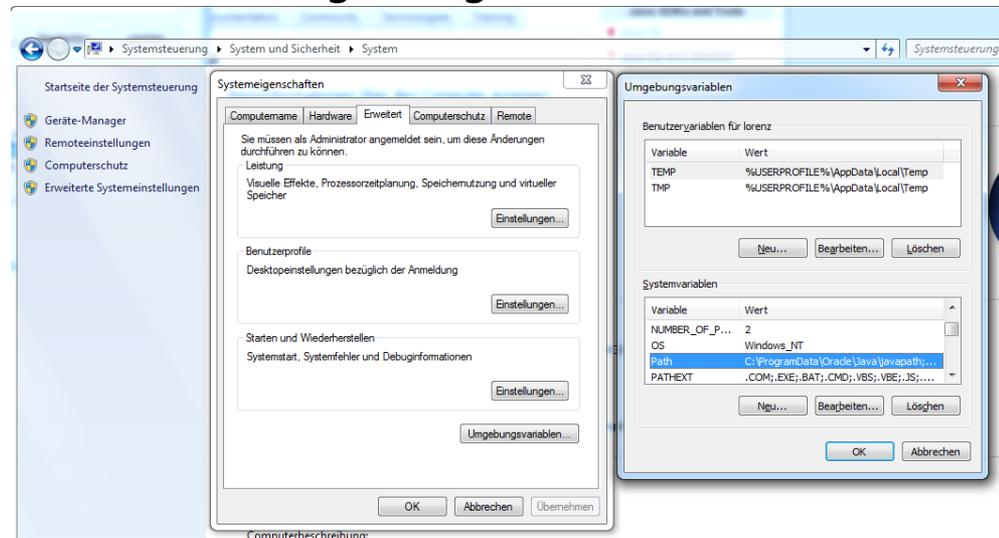
- Java Standard Edition SE
 - Java Plattform für Desktop und einfache Serveranwendungen
 - Aktuell (Stand: 19.10.2016) Java 8 Update 111 (Java SE 8u111)
 - Kostenlos auf
<http://www.oracle.com/technetwork/java/javase/downloads/index.html>
- Java Enterprise Edition EE
 - Java Plattform für komplexere Server und Netzwerkanwendungen
- Java Micro Edition ME
 - Reduzierte Java Plattform für mobile Geräte und eingebettete Systeme
 - Oracle-Lizenz erforderlich

Schritt 1: JDK Standard Edition SE downloaden

- Auf <http://www.oracle.com/technetwork/java/javase/downloads/index.html>
- Richtige Plattform auswählen (Linux, Mac, Solaris, Windows)
- Gespeicherte **jdk.exe** ausführen und installieren

Schritt 2: Ausführungspfad setzen

- Systemsteuerung -> System -> Erweiterte Systemeinstellungen auswählen -> Unter dem Reiter „Erweitert“ die Schaltfläche *Umgebungsvariablen* anklicken
- Pfad des JDK\bin eintragen. Bsp.:
 - C:\Program Files\Java\jdk1.8.0_111\bin
 - Trennung mit ;



Installationshinweise für OS X unter:

- https://docs.oracle.com/javase/8/docs/technotes/guides/install/mac_jdk.html

Installationshinweise für Linux (Ubuntu) unter:

- https://wiki.ubuntuusers.de/Java/Installation/Oracle_Java/Java_8

Schritt 3: Installation testen

- In Konsole eingeben: `java -version`
- Ergebnis: „`java version 1.8.0_102`“

```
Eingabeaufforderung
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. Alle Rechte vorbehalten.

C:\Users\schauer>java- version
Der Befehl "java-" ist entweder falsch geschrieben oder
konnte nicht gefunden werden.

C:\Users\schauer>java -version
java version "1.8.0_102"
Java(TM) SE Runtime Environment (build 1.8.0_102-b14)
Java HotSpot(TM) 64-Bit Server VM (build 25.102-b14, mixed mode)

C:\Users\schauer>_
```

3 Schritte sind zu durchlaufen:

- Erstellen des Quellcodes
- Kompilieren: Übersetzen des Quellcodes (.java) in Bytecode (.class)
- Starten des Programms durch Übergabe des Bytecodes an den Interpreter JVM

3 Schritte sind zu durchlaufen:

- **Erstellen des Quellcodes**
- Kompilieren: Übersetzen des Quellcodes (.java) in Bytecode (.class)
- Starten des Programms durch Übergabe des Bytecodes an den Interpreter JVM

Der Quellcode kann mit jedem beliebigen Texteditor erzeugt werden.

- Bsp.: Notepad++ (Windows <https://notepad-plus-plus.org/download/v6.8.3.html>)
- Bsp.: Geany (für Linux)
- Java Datei anlegen:
 - Bsp.: NameOhneUmlaute.java
 - Entspricht auch dem Klassennamen! => *public class NameOhneUmlaute*
 - Konvention: Großer Anfangsbuchstabe für Klassen (also auch für Java-Dateien)
 - Bei Windows: Dateiendung einblenden, um .java statt .txt zu erzeugen

```
aufgabe4.txt x Student.java x
1 // Package Deklaration
2 package java.fuer.anfaenger;
3
4 // Import externe Klassen
5 import java.util.*;
6
7 // Klassendefinition
8 public class Student{
9
10     // Instanzvariablen
11     private int alter;
12     private String name;
13
14     // Konstruktor
15     public Student(String name){
16         this.name = name;
17     }
18
19     // Methode
20     public void studieren(int jahre){
21         System.out.println(name + " studiert schon "+jahre+" Jahre");
22     }
23 }
```

3 Schritte sind zu durchlaufen:

- Erstellen des Quellcodes
- **Kompilieren: Übersetzen des Quellcodes (.java) in Bytecode (.class)**
- Starten des Programms durch Übergabe des Bytecodes an den Interpreter JVM

In der Konsole (cmd bei Windows)

- Wechseln ins Verzeichnis der Java-Datei
- Kompilieren mit dem Befehl: **javac** <NameDerDatei>.java

```
cmd Eingabeaufforderung
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. Alle Rechte vorbehalten.

C:\Users\schauer>cd Documents

C:\Users\schauer\Documents>javac Test.java

C:\Users\schauer\Documents>
```

3 Schritte sind zu durchlaufen:

- Erstellen des Quellcodes
- Kompilieren: Übersetzen des Quellcodes (.java) in Bytecode (.class)
- **Starten des Programms durch Übergabe des Bytecodes an den Interpreter JVM**

Ausführen in der Konsole:

- mit dem Befehl: `java <Name>`
 - **Hinweis:** ohne Endung .class

```
C:\> Eingabeaufforderung
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. Alle Rechte vorbehalten.

C:\Users\schauer>cd Documents
C:\Users\schauer\Documents>javac Test.java
C:\Users\schauer\Documents>dir/W
Datenträger in Laufwerk C: ist OS
Volumeseriennummer: 8074-517E

Verzeichnis von C:\Users\schauer\Documents

[.]                               [..]
Test.class                         Test.java
                                   532 Bytes
                                   3 Verzeichnis(se), 415.401.893.888 Bytes frei

C:\Users\schauer\Documents>java Test
Hello World!

C:\Users\schauer\Documents>
```

Einfache Texteditoren sind ausreichend aber umständlich.

- Daher: Unterstützung beim Programmieren durch IDEs

Eine IDE (*Integrated Development Environment*) bietet i.d.R.:

- Texteditor
- Compiler bzw. Interpreter
- Linker
- Debugger
- Quelltextformatierungsfunktion

Vorteile:

- bietet viel mehr Features und Funktionen als einfache Editoren.
 - Syntax-Highlighting, Auto-Vervollständigung, Such-Funktionen, usw.
- spart viel Zeit bei der Programmierung.
- erkennt Tipp- sowie Syntaxfehler.
- ermöglicht Projektmanagement und Teamarbeit.

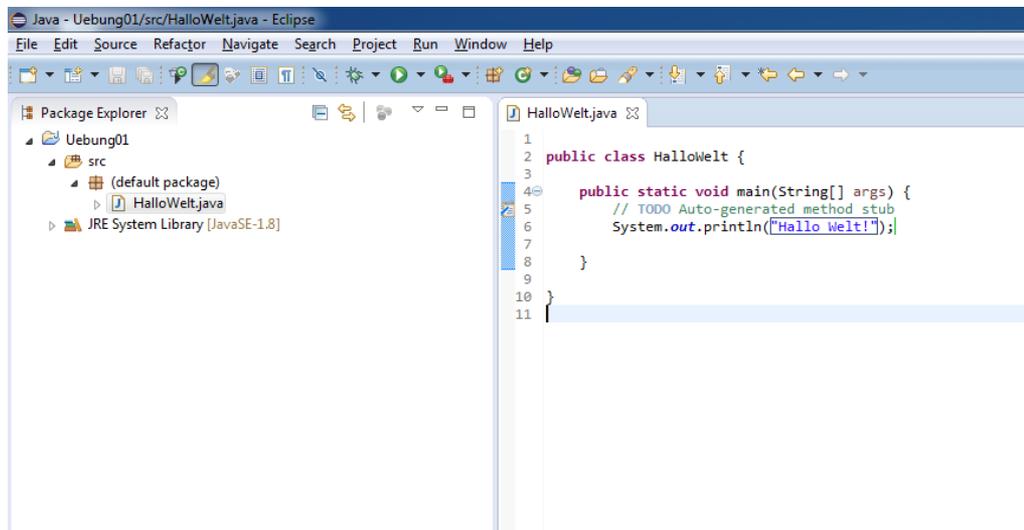
Kostenlose Java Entwicklungsumgebung:

- Eclipse IDE: <https://www.eclipse.org/home/index.php>
 - Wird sehr häufig verwendet – auch in diesem Kurs!
- NetBeans IDE: <https://netbeans.org/>
 - Hauptsächlich für Java entwickelt, aber auch Unterstützung für andere Sprachen
- BlueJ: <http://www.bluej.org/>
 - Speziell für Ausbildungszwecke
 - Grafische Darstellung von Objekten
- IntelliJ IDEA: <https://www.jetbrains.com/idea/>
 - Gebräuchlich für Mobile Development
 - Freie Community-Edition und Studentenversion

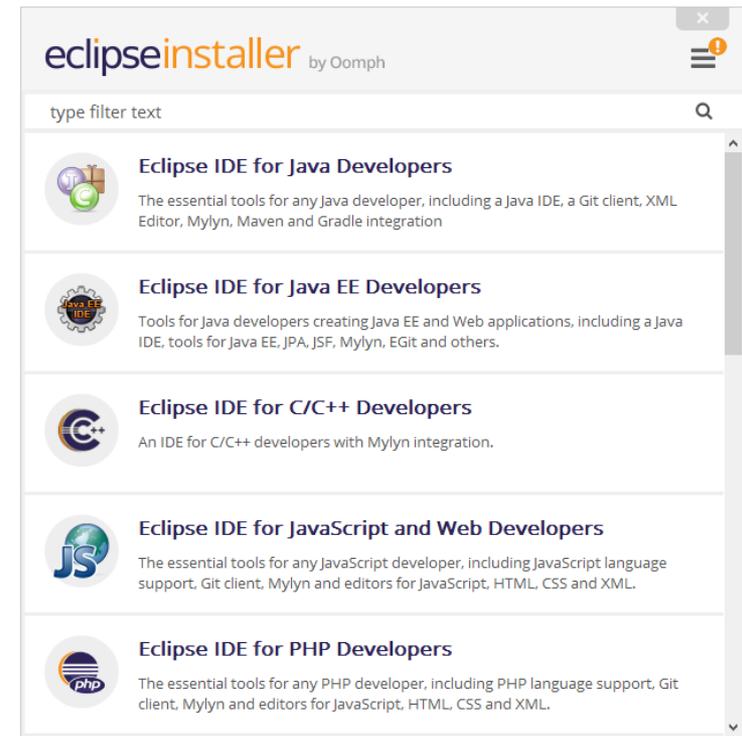


Kurze Einführung in Eclipse

- Download Eclipse Installer: <https://www.eclipse.org/downloads/>
- .exe öffnen und „richtiges“ Eclipse auswählen:
 - *Eclipse IDE for Java Developers*
- Eclipse installieren und dann starten
- Workspace festlegen (Ordner für Projekte)
- Erstes Projekt erstellen...
 - Hallo Welt!



```
1 public class HalloWelt {
2
3
4     public static void main(String[] args) {
5         // TODO Auto-generated method stub
6         System.out.println("Hallo Welt!");
7     }
8 }
9
10
11 }
```



Was haben wir gelernt?

- Geschichte und Vorteile von Java
- Compiler und Interpreter
- Erstes Java Programm schreiben, kompilieren und ausführen
- Entwicklungsumgebungen (Eclipse)

Wie geht's weiter?

- Klassen und Objekte
- Grundstruktur einer Java Klasse
- Variablen und Datentypen
- ...