HAW Hamburg
Fakultät TI
Technik und Informatik

Programmiermethodik 1
Programmiertechnik

Einführung

Herzlich Willkommen in der Informatik der HAW Hamburg!

Ziel der Veranstaltung

Spaß am
Programmieren lernen!

Ziel der Veranstaltung

- Die Vorgehensweise bei der Programmentwicklung *(„Softwareentwicklung“)* vermitteln und einüben
- Konzepte und Sprachmittel einer aktuellen Programmiersprache (*Java*) vermitteln und einüben

Motivation

- Basisaufgabe der Informatik:

Die Welt mit formalen Methoden beschreiben und diese zur Problemlösung einsetzen.

Voraussetzungen

- Keine Programmierkenntnisse nötig, aber dafür folgendes:
  - Erfahrungen im elementaren Umgang mit einem Computer
    (Programme starten, Dateien mit Texteditor bearbeiten und speichern, Internetbrowser zum Dateidownload verwenden)
  - Hohe Motivation
  - Fähigkeit, systematisch und gewissenhaft zu arbeiten
  - Bereitschaft, ein Buch oder Online-Dokumentation zu lesen
  - Bereitschaft zum intensiven Üben

Was ist Programmieren?

- Computerprogramme schreiben.

- Genauer (Wikipedia: Programmierung):

Programmierung [...] bezeichnet die Tätigkeit, Computerprogramme zu erstellen. Dies umfasst vor Allem die Umsetzung (Implementierung) des Softwareentwurfs in Quellcode sowie – je nach Programmiersprache – das Übersetzen des Quellcodes in die Maschinensprache, meist unter Verwendung eines Compilers.

Was ist ein Computerprogramm?

- Eine Reihenfolge von Befehlen, die dem Computer sagen, was er machen soll.

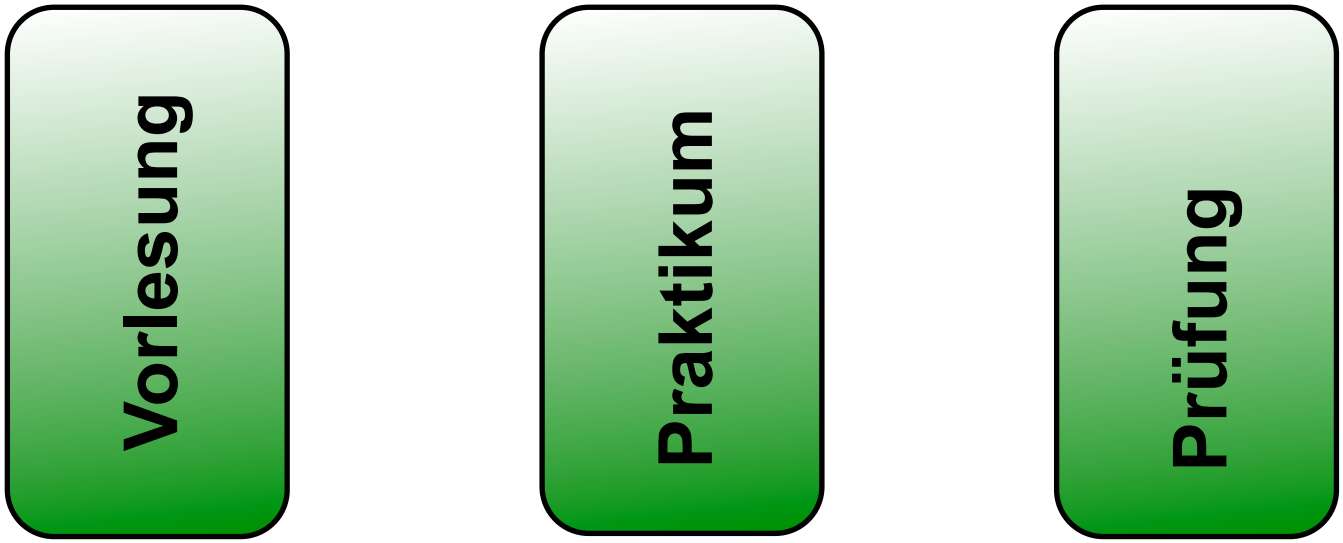
Umfrage

https://users.informatik.haw-hamburg.de/~abo781/gcrs/vote.html

Organisation

Organisation

- Drei Säulen



Organisation: Vorlesung

- keine Anwesenheitspflicht
- statt 28 x 3h: 21 x 4h
  - daher entfallen die letzten Termine im Semester

Organisation: Praktikum

- Anwesenheitspflicht
- Bearbeitung der Aufgaben in 2er-Teams
  - Anmeldung bereits erfolgt (StiSys)
- Abnahme
  - Vorstellung der Lösung zum Praktikumstermin
  - erfolgreiche Abnahme aller Praktikumsaufgaben ist Prüfungsvoraussetzung

Organisation: Praktikum

Anforderungen an abgegebene Lösungen

- Bearbeitung aller Teilaufgaben
- keine Kompilierfehler, keine Compiler-Warnungen
- später zusätzlich: Code-Konventionen
- beide Teammitglieder können gesamte Lösung erläutern
- Einhalten der Code-Konventionen
- Präsentation auf dem Poolrechner
- Entwicklungsumgebung: freigestellt, aber Support nur für Eclipse

Organisation Praktikum

- Abnahme in den Praktika jeweils
  - ein Professor + ein wissenschaftlicher Mitarbeiter
- bei uns
  - Prof. Michael Schäfers
  - Prof. Axel Schmolitzky
  - Prof. Philipp Jenke
  - Norbert Kasperczyk (wissenschaftlicher Mitarbeiter)
- Sprechstunde
  - nach Vereinbarung (per Mail oder im Praktikum)

Organisation: Prüfung

- zwei Prüfungen am Semesterende
  - Klausur (schriftlich, Papier)
  - Rechnerprüfung (Programmieren am Rechner)

- zum Üben: Mid-Term
  - Probeklausr in der Mitte des Semesters
  - Teilnahme freiwillig
  - kein Einfluss auf die Note am Semesterende

Organisation: EMIL

- E-Learning-Plattform der HAW Hamburg
- zentraler Anlaufpunkt für alle Informationen und Materialien
- URL: *http://www.elearning.haw-hamburg.de*
  - Login: HAW-Login
- Unser Lernraum
  - URL: *http://www.elearning.haw-hamburg.de/course/view.php?id=10104*
  - Suche: Jenke, Programmiermethodik 1
  - Registrierung für Lernraum
    - Selbsteinschreibeschlüssel. PM1PTSS15

Organisation: Tutorium

- Angebot: wöchentliches Tutorium
- zwei Tutoren: Studierende aus höherem Semester
  - Florian Heiwig
  - Clemens Raßbach
- Inhalte
  - zusätzliche Übungsaufgaben
  - Hilfe bei Fragen zu Praktikumsaufgaben
  - Unterstützung beim Ankommen an der Hochschule
    - z.B. Selbstorganisation, richtig Lernen, ...

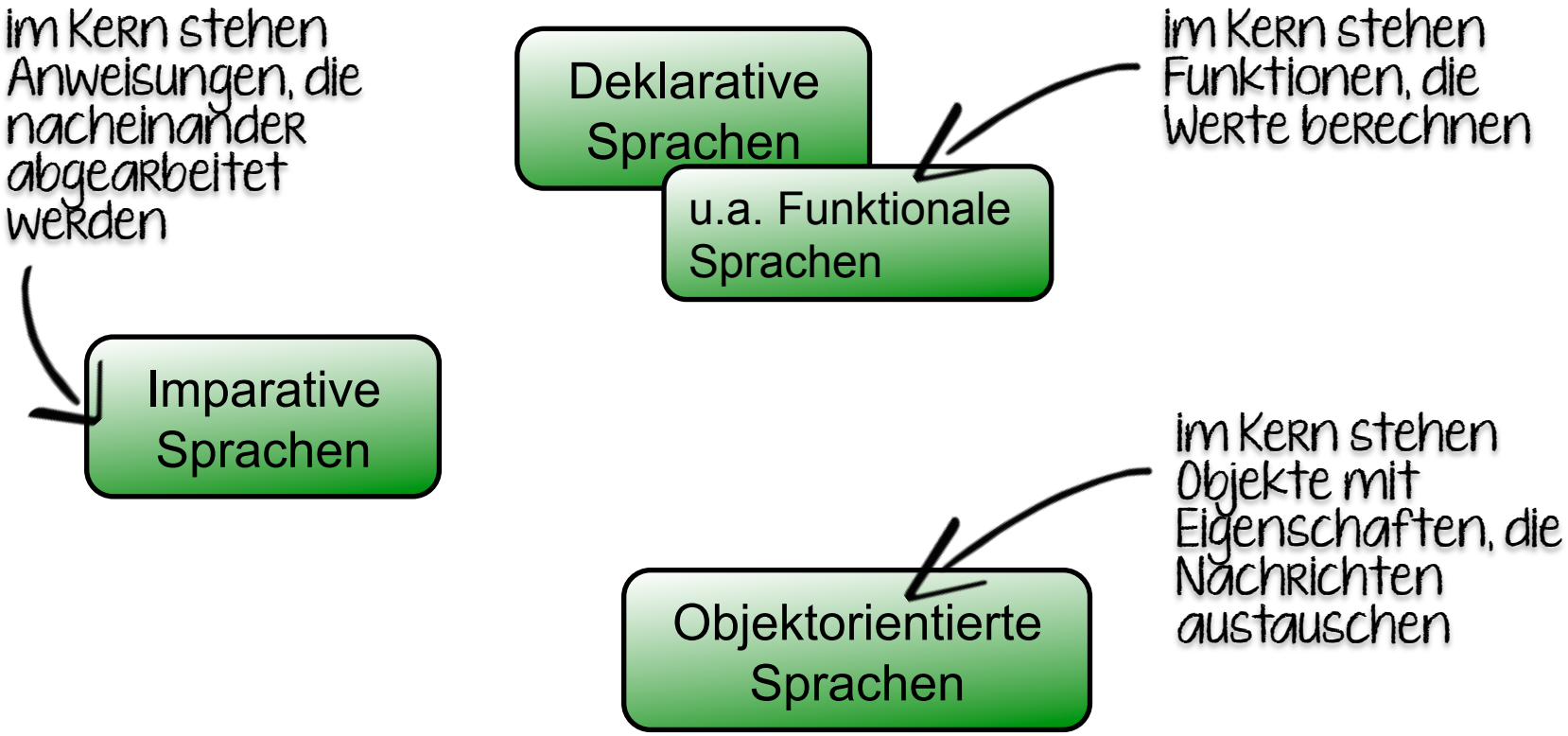
Programmiersprache

Es gibt viele Programmiersprachen

HAW Hamburg
Fakultät TI
Technik und Informatik

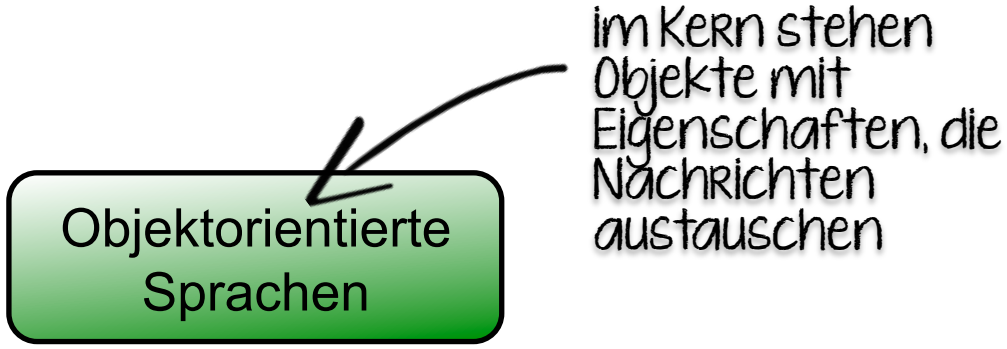
Höhere Programmiersprachen

- Kategorisierung anhand der zentralen Programmierparadigmen



Was nehmen wir

- Wir nehmen Java



- Warum Java? Java ist …
  - modern
  - einfach
  - weit verbreitet
  - für viele verschiedene Betriebssysteme verfügbar
  - kostenlos
  - für alle Arten von Problemen flexibel einsetzbar
  - im Department Informatik in vielen Praktika im Einsatz

Java

- Kostenlos erhältlich (für alle gängigen Betriebssysteme)
- aktuelle Version: Java 8

- Download:

http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

- JDK vs. JRE
  - Achtung: Wir benötigen JDK (Java Development Kit)
  - nicht JRE (Java Runtime Environment)

Literaturempfehlungen

- Reinhard Schiedermeier: Programmieren mit Java, 2. Auflage, Pearson Studium, 2010

  Sehr gut lesbares Lehrbuch zur Einführung der grundlegenden Konzepte

- Kathy Sierra, Bert Bates: Java von Kopf bis Fuß, 2. oder 3. Auflage, O'Reilly

  Spielerische Einführung in Java und objektorientierte Konzepte

- Philip Ackermann: Schrödinger programmiert Java, Galileo Computing

  Ähnlich dem vorherigen Buch, ebenfalls "etwas andere" Didaktik

- Christian Ullenboom: Java ist auch eine Insel, 10. Auflage, Galileo Computing, 2012

  Ausführliches Standardwerk zum Lernen und Nachschlagen mit vielen Beispielen Online-Version / Kostenloser Download unter http://openbook.rheinwerk-verlag.de/javainsel/

- Offizielle JAVA-Referenz: *http://www.oracle.com/technetwork/java/index.html*

  Für das Praktikum ist das Development Kit (JDK) der Java Standard Edition (SE) nötig

Was umfasst „Programmieren“?

- Klären, welches Problem das neue Programm überhaupt lösen soll
  - „Anforderungsanalyse“: Was soll das Programm genau tun?
- Aufbau des neuen Programms überlegen
  - „Entwurf“: Wie soll das Programm strukturiert sein?
- Programmcode („Sourcecode“) in einer Programmiersprache schreiben
  - „Implementierung“
- Sicherstellen, dass das Programm zuverlässig funktioniert
  - „Test“
- Dokumentieren aller Schritte!

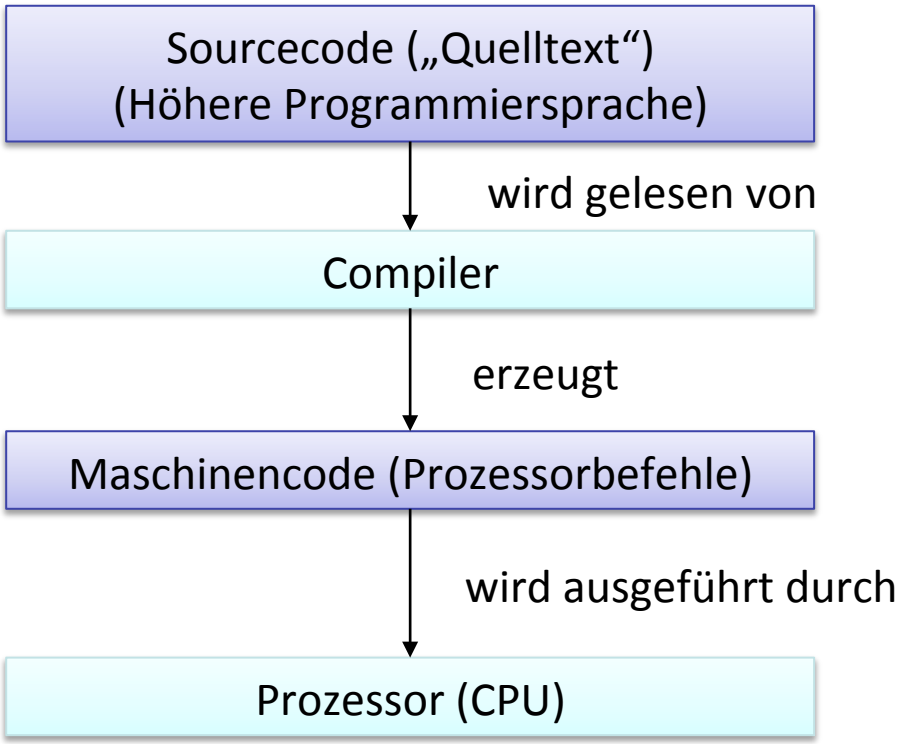
Beispielprogramm "Hallo Welt!"

- Anforderungsanalyse: *„Hallo Welt!“* ausgeben
- Entwurf: Einfache Ausgabeanweisung verwenden
- Implementierung: Java-Sourcecode
  - Datei: *HalloWelt.java*

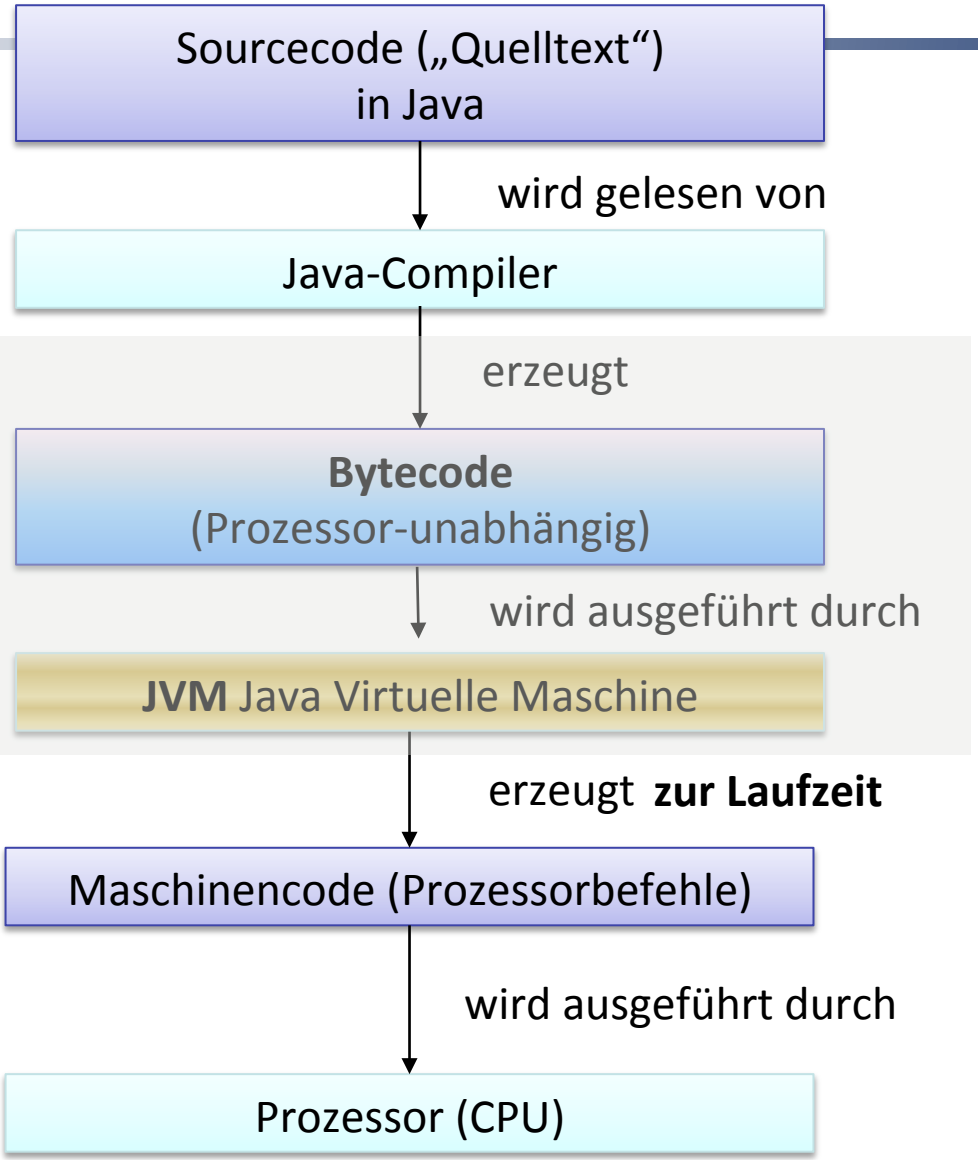
```
1 package einfuehrung;
2
3 /**
4  * Gibt den Text "Hallo Welt!" auf der Konsole aus.
5  */
6 public class HalloWelt {
7   public static void main(String[] args) {
8     System.out.println("Hallo, Welt!");
9   }
10 }
```

Wie kommen wir vom Quellcode zum laufenden Programm?

Funktion eines Compilers



Spezialfall JAVA-Compiler



Beispielprogramm "Hallo Welt!"

- Sourcecode in Datei speichern

  HalloWelt.java in *<Verzeichnis>*
- In Verzeichnis wechseln

  cd <Verzeichnis>
- Java-Compiler aufrufen (Sourcecode → Bytecode):

  javac HalloWelt.java
  - Ergebnis: neue Datei *HalloWelt.class*
- Java-VM *HalloWelt.class* (→ den Bytecode) ausführen lassen

  java Summe

Entwicklungsumgebung

Entwicklungsumgebung: Eclipse

- für größere Programme ist ein Editor hilfreich
  - Verwalten mehrerer Programm-Dateien
  - Kompilieren
  - Ausführen
  - Unterstützung bei der Programmierung (z.B. Syntax-Highlighting)

- Lösung: (Integrierte) Entwicklungsumgebung
  - auch Integrated Development Environment oder IDE
  - wir verwenden Eclipse
  - aktuelle Version: Eclipse Luna (4.4)

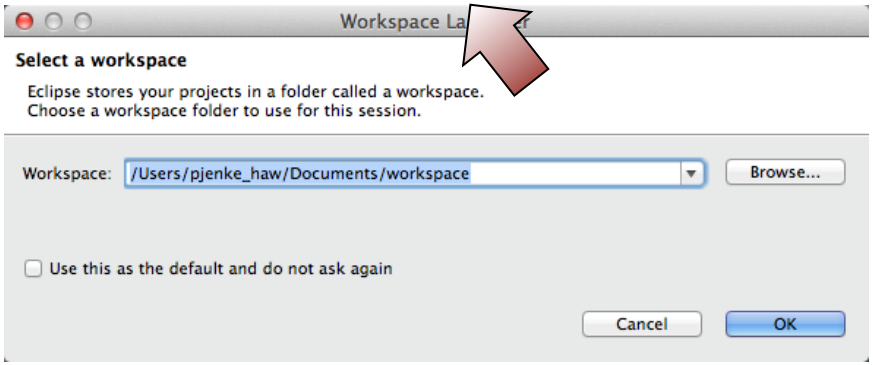
Installation und Start

- Download
  - *http://www.eclipse.org/*
  - Version: Eclipse IDE for **Java** Developers
- Installation = Entpacken des Pakets
  - ggf. Verschieben des Verzeichnisses *eclipse* an einen zentralen Ort
- Starten
  - Wechsel in das e*clipse*-Verzeichnis
  - Starten des Programms *eclipse(.exe) (Windows)*

Erste Konfiguration

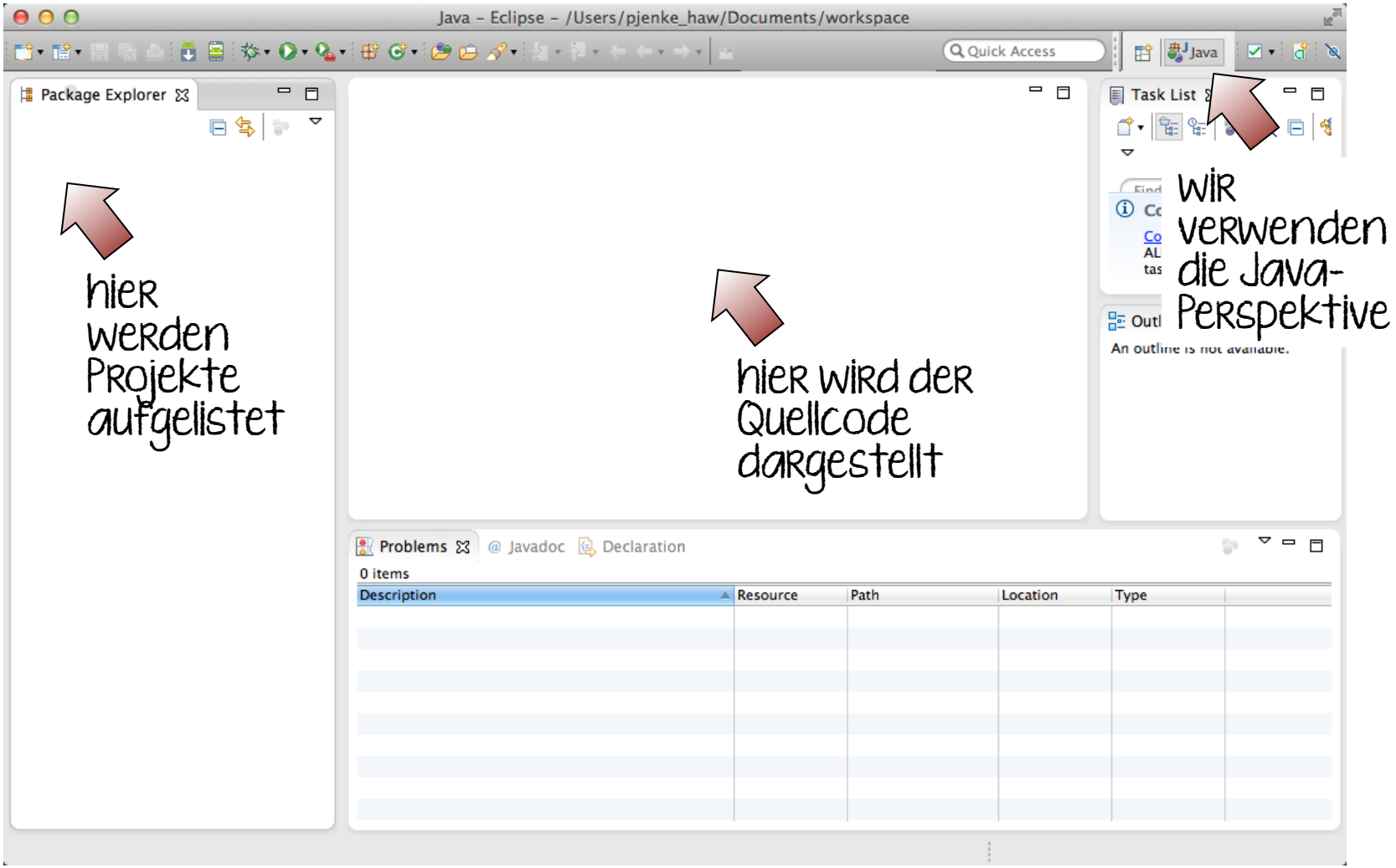
Achtung: Die Screenshots sehen je nach Version etwas anders aus!

- Setzen des Workspace

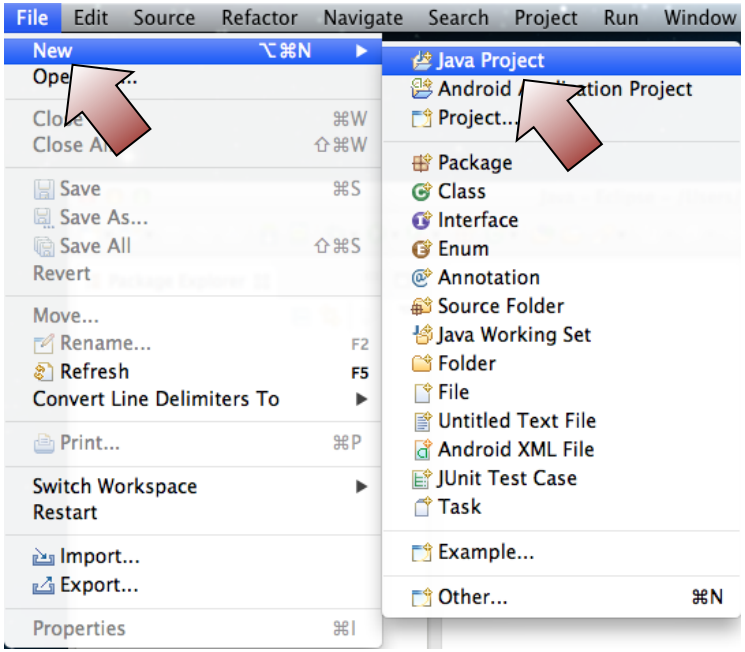


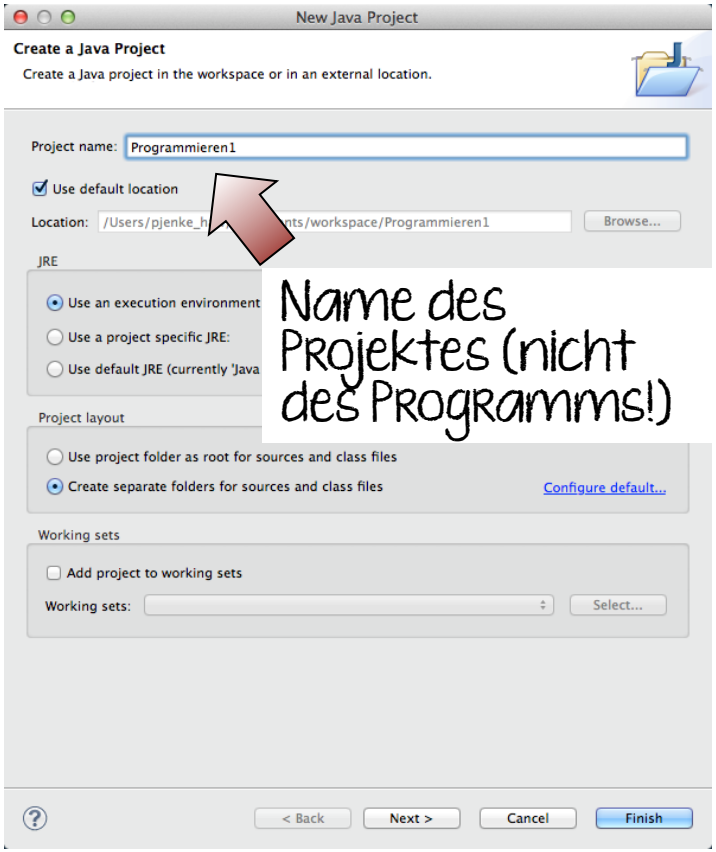
- in diesem Verzeichnis werden standardmäßig die Projekte abgelegt

Die Perspektive



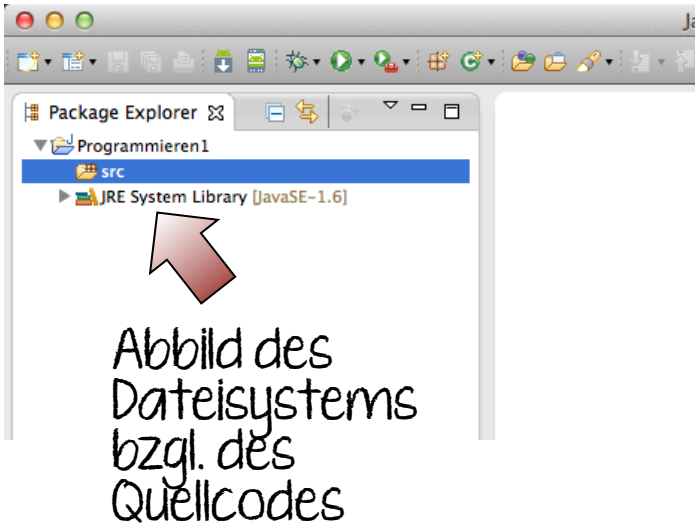
Neues Projekt Anlegen



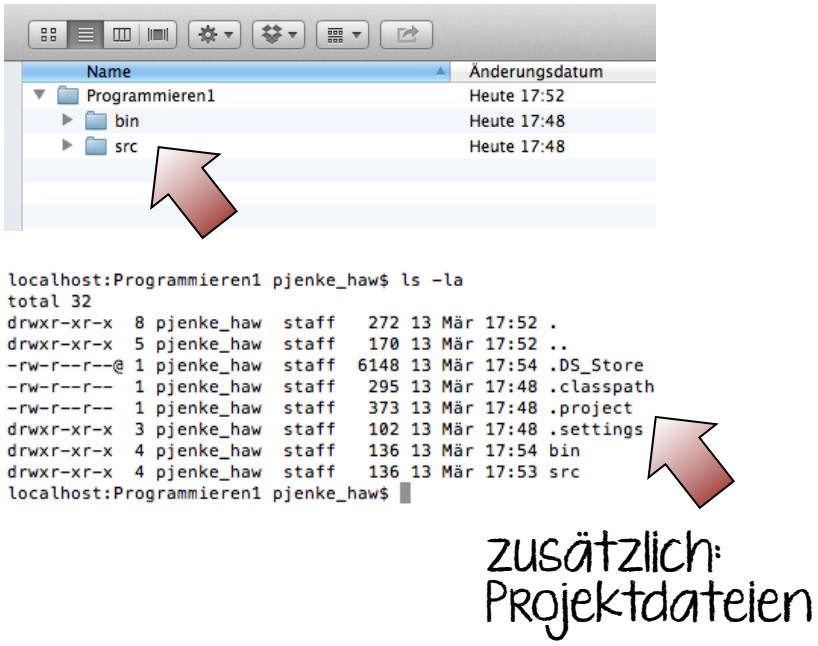


Neues Projekt

- in Eclipse

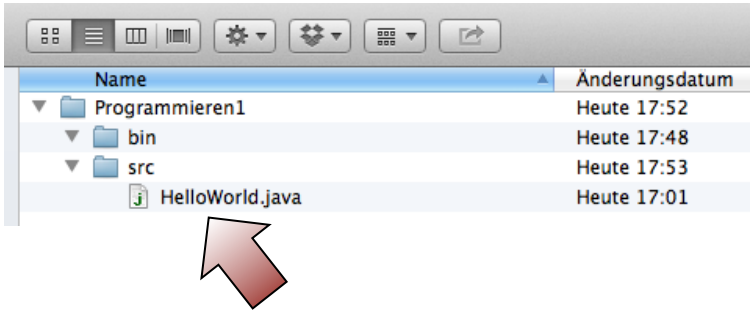


> im Dateisystem

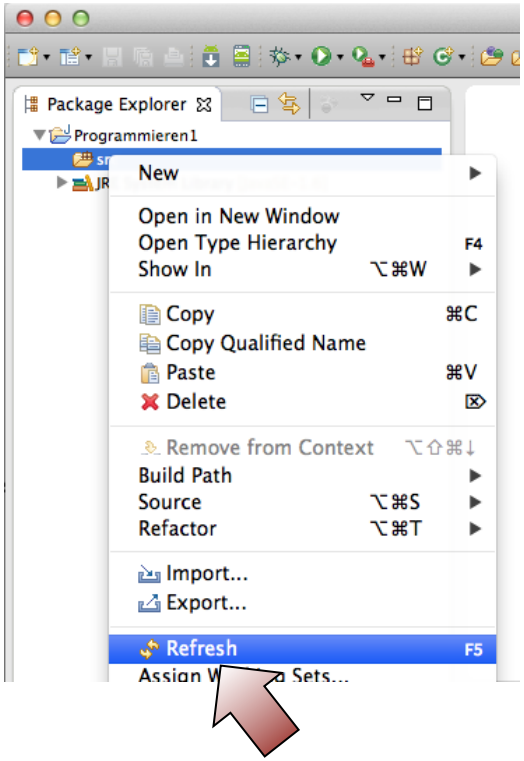


Einfügen von HalloWelt.java

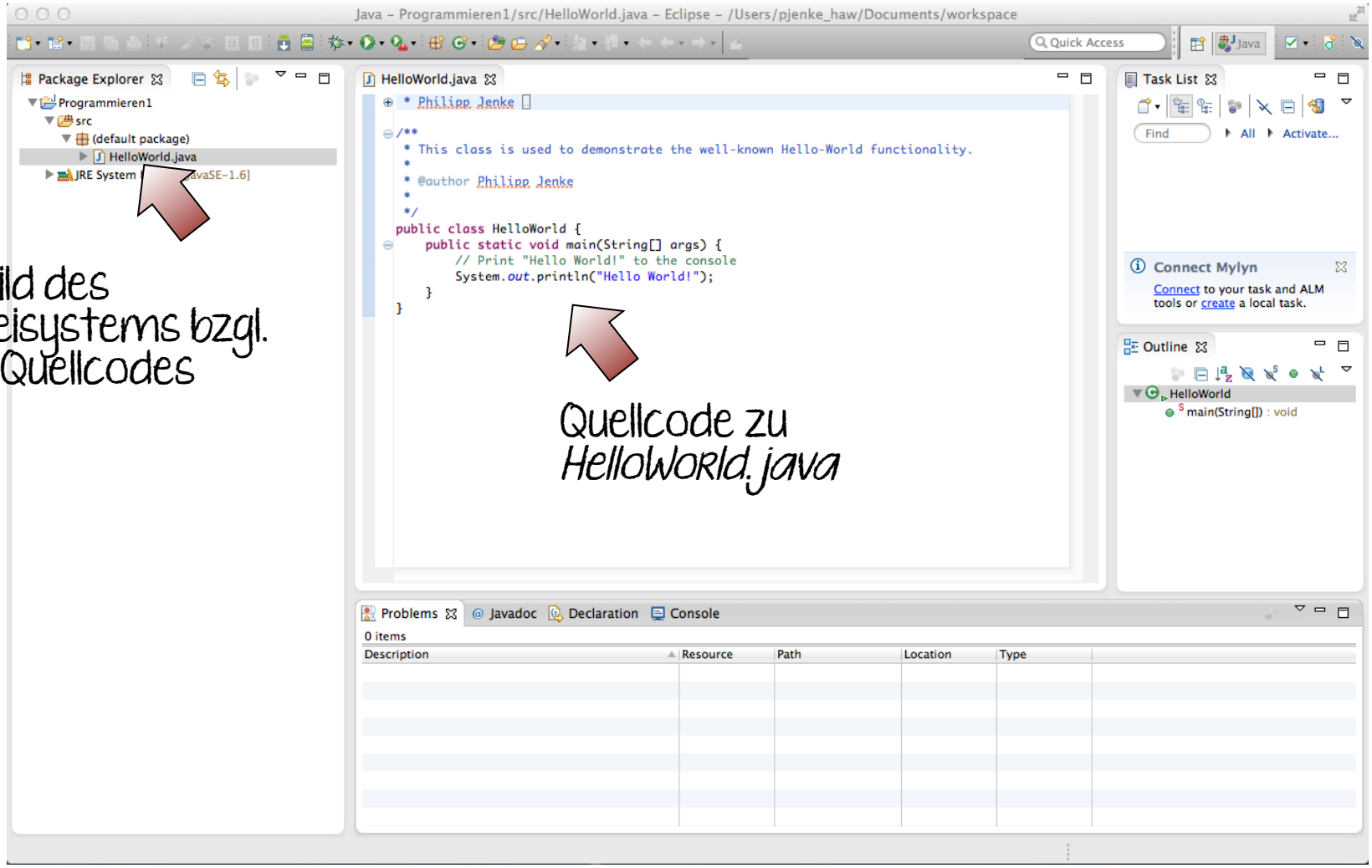
- im Dateisystem



- in Eclipse
  - Rechtsklick auf den Ordner *src*



HelloWorld in Eclipse

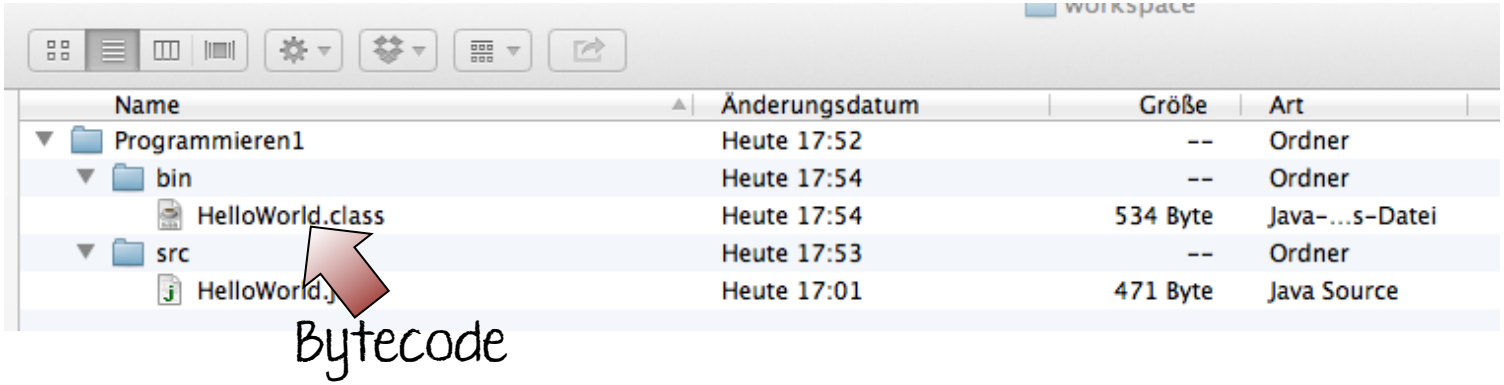


Abbild des Dateisystems bzgl. des Quellcodes

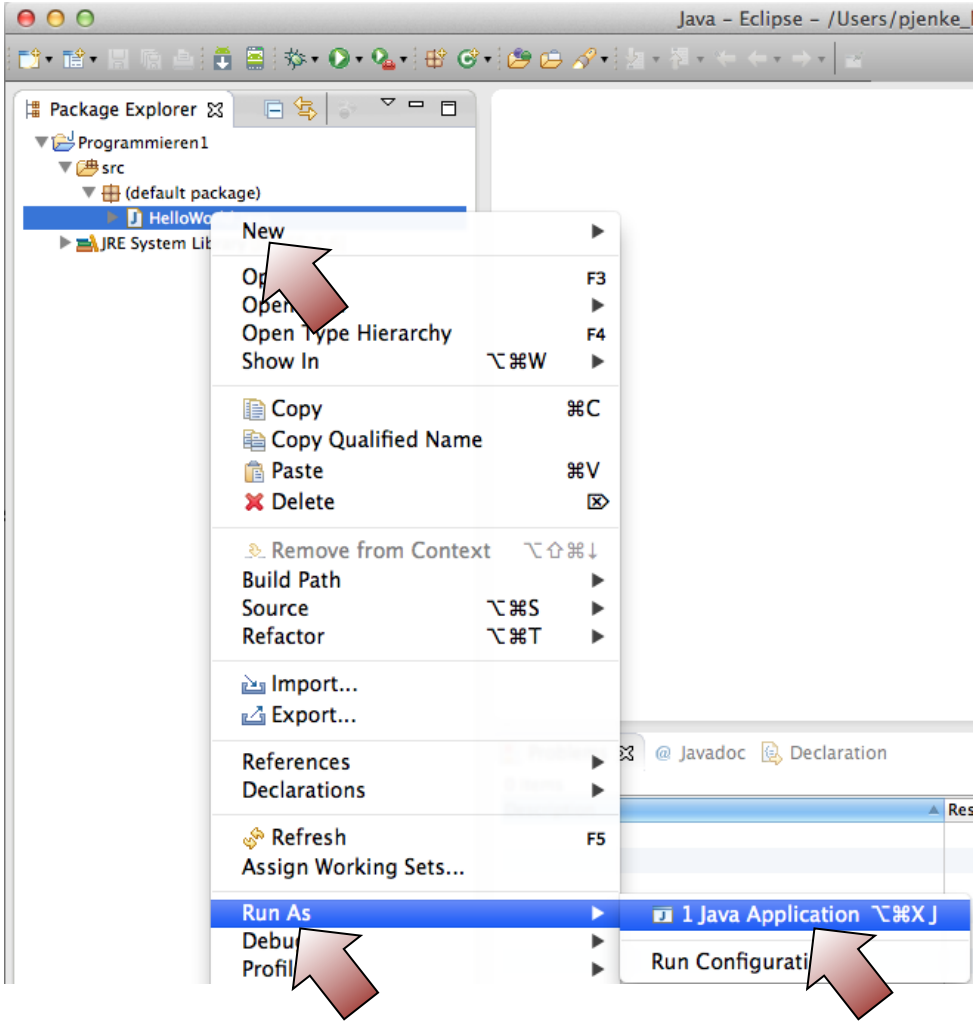
Quellcode zu HelloWorld.java

Kompilieren in Eclipse

- wird von Eclipse automatisch gemacht
- Blick in das Dateisystem:



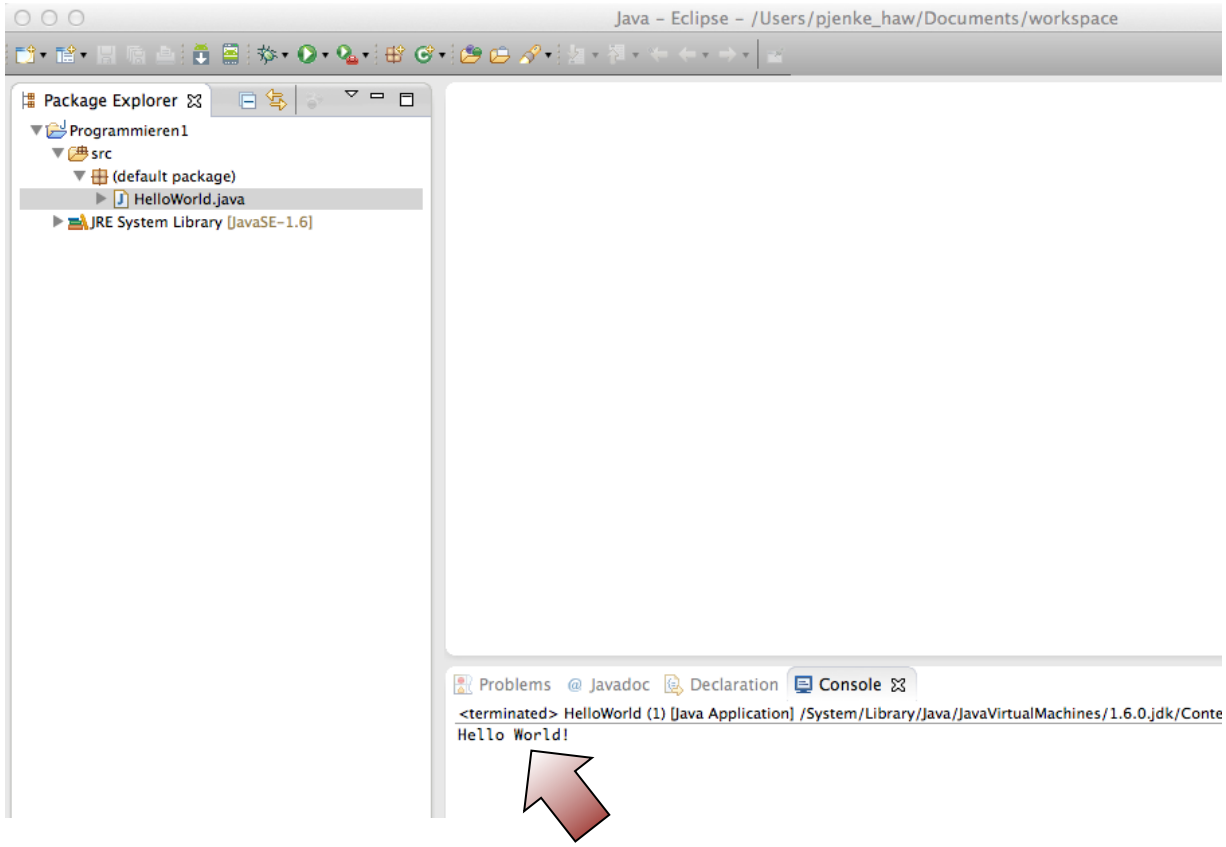
Ausführen eines Programms



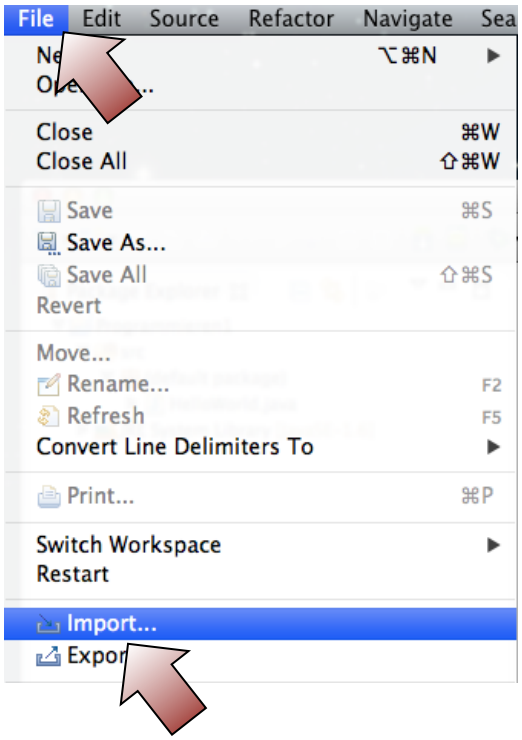
- Rechtsklick auf *HelloWorld.java*

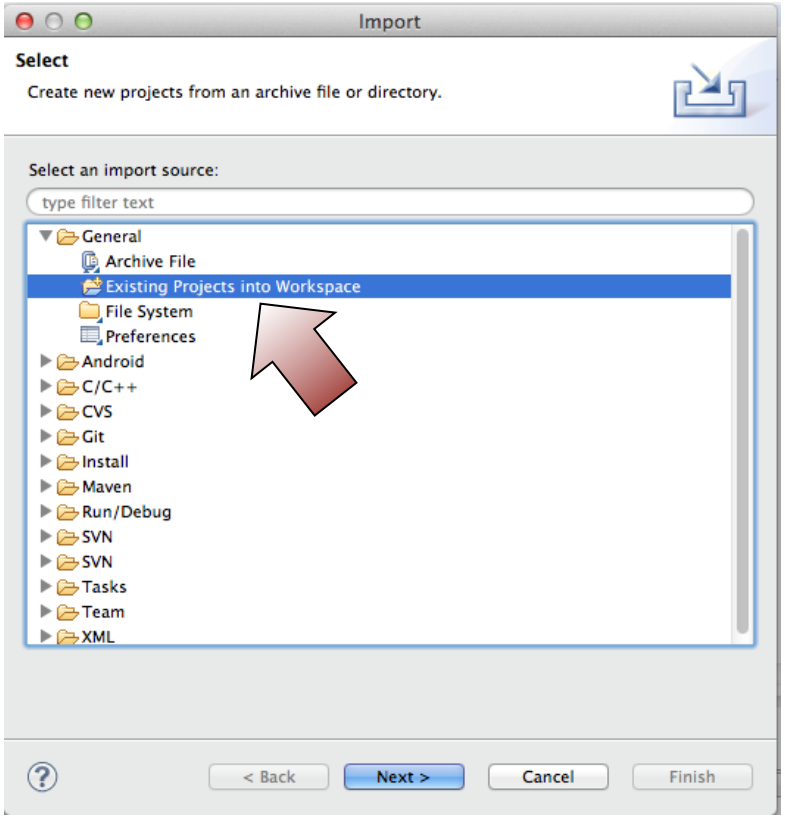
Ausführen eines Programms

- Konsolenausgabe



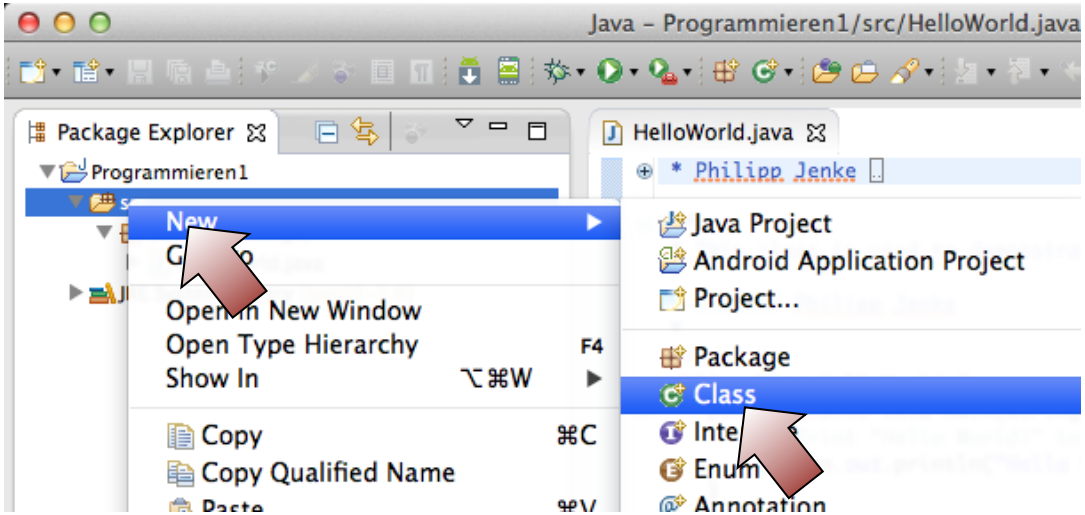
Importieren eines Projektes



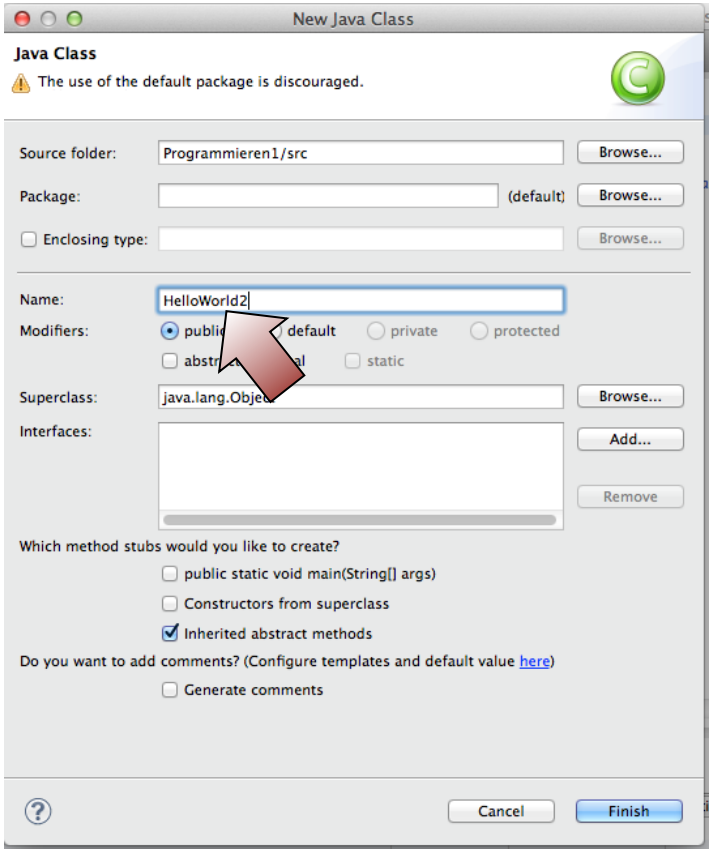


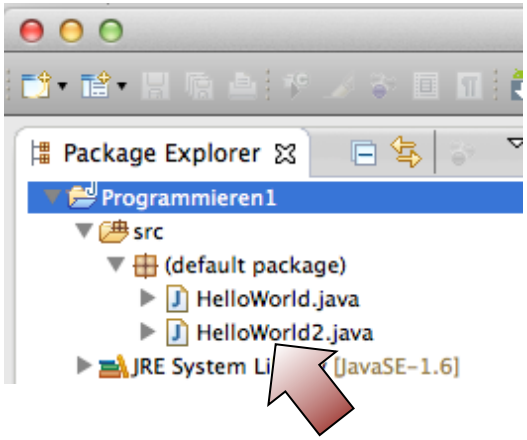
Anlegen eines neues Programms

- Rechtsklick auf den Ordner *src*
- Bei uns zunächst: Programm = Klasse (engl. Class)



Anlegen Neues Programm





Einschub: Package

- jede Java-Klasse sollte einem Package zugewiesen werden
- Packages können frei gewählt werden
- Beispiel: `variablen;`
  - Zuweisung im Quellcode (.java): `package variablen;`
  - in Eclipse: steht im Package-Explorer
- Package muss sich auch im Verzeichnisbaum wiederspiegeln

Weiteres Beispiel

- "Berechne die Summer aller Zahlen von 1 bis n."

Anforderungsanalyse

- Input:
  - Eine beliebige natürliche Zahl **`n`**
- Output:
  - Die Summe aller natürlichen Zahlen von 1 bis n:
    1 + 2 + 3 + … + **`n`**

- *Beispiel:*
  - *Input:* **`n`** *= 4*
  - *Output: 10*

Entwurf

Algorithmus als „**Kochrezept**“:

- Variablen (Speicherplätze) als „Zutaten“
  a. `zahl` Input
  b. `ergebnis` Summe der bisher addierten Zahlen
  c. `zaehler` Die Zahl, die jeweils an der Reihe ist (wird hochgezählt)
- Anweisungen (Befehle) als „Zubereitung“
  1. Input `n` festlegen
  2. `ergebnis` den Wert 0 zuweisen (bisher noch nichts addiert)
  3. `zaehler` den Wert 1 zuweisen (erste Zahl)
  4. Wiederholen, solange `zaehler` ≤ `zahl` ist …
     a) `ergebnis` um den Wert von `zaehler` erhöhen
     b) `zaehler` durch die nächste Zahl ersetzen, also „hochzählen“
  5. Die Summe `ergebnis` ausgeben

Implementierung (in Java): Sourcecode


```java
package einfuehrung;

/**
 * Berechnet die Summe aller nat. Zahlen 1 .. n.
 */
class Summe {

  public static void main(String[] args) {
    // Variablen (Zutaten)
    int zahl;
    int ergebnis;
    int zaehler;

    // Anweisungen (Zubereitung)
    zahl = 6;
    ergebnis = 0;
    zaehler = 1;

    while (zaehler <= zahl) {
      ergebnis = ergebnis + zaehler;
      zaehler = zaehler + 1;
    }

    System.out.println(ergebnis);
  }
}
```

Was ist was?

- `class`
  - kündigt ein neues Programm an
  - Name Summe folgt direkt danach
  - Programmname und Dateiname (*Summe.java*) müssen gleich sein!
- geschweifte Klammern **{**
  - grenzen zusammen gehörende Quelltext-Abschnitte ab („Blöcke“)
  - Klammern nach der `class`-Zeile umfassen das gesamte Programm.
- `main`
  - markiert den Start der eigentlichen Anweisungen
  - wird auch als „Einsprungpunkt“ bezeichnet, weil hier die Ausführung des Programms beginnt:

```
public static void main(String[] args)
```

Was ist was?

- eigentliches Programm
  - wieder zwischen geschweiften Klammern (Ende der `main`-Zeile)
  - besteht aus Anweisungen (meist eine Zeile)
- Anweisungen
  - nun folgen verschieden Anweisungen
  - enden mit einem Semikolon (;)
  - z.B. `System.out.println` = Ausgabe auf der Konsole

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

Ausblick

Wie geht es weiter?

Ausblick

- Bezeichner
- Variablen
- Arithmetische Ausdrücke
- Zahlentypen
- Wahrheitswerte und Bedingungen
- Schleifen
- einfache Datentypen
- Klassen
- Methoden
- Konstruktoren

- unveränderliche Klassen
- statische Variablen
- Aufzählungstypen
- Testen
- Wrapper, Strings
- Arrays
- Interfaces
- Vererbung
- Rekursion
- Ausnahmebehandlung
- Collections

Zusammenfassung

- Einführung
- Programmieren
- Organisation
- Java
- Erstes Programm
- Eclipse