

Java-Einführungskurs Informatik II (D-ITET)

Vincent Becker, vincent.becker@inf.ethz.ch



Was haben wir heute vor?

- Vorbereitung auf die Übungen zu Informatik II
 - Vorstellung des Teams
 - Organisatorisches
- Theorie
 - Java-Technologie und Sprache
- Praktisches
 - Erstes Programm auf der Konsole ausführen
 - Eclipse
 - Debugging
 - Testen
 - Javadocs

institute for
pervasive computing



RESEARCH GROUP FOR
Distributed Systems




Übungsgruppen

- Mittwochs, 13 – 14 Uhr
- Die Gruppeneinteilung läuft über CodeExpert: <https://expert.ethz.ch/enroll/SS19/ifee2>
- Anmeldung auf Codeboard.io ist notwendig: <https://codeboard.io>
- Anwesenheit in den Übungsgruppen ist wichtig
- Ebenso die Bearbeitung der Aufgaben

Fragen & Interaktion!!!

Bonusaufgaben

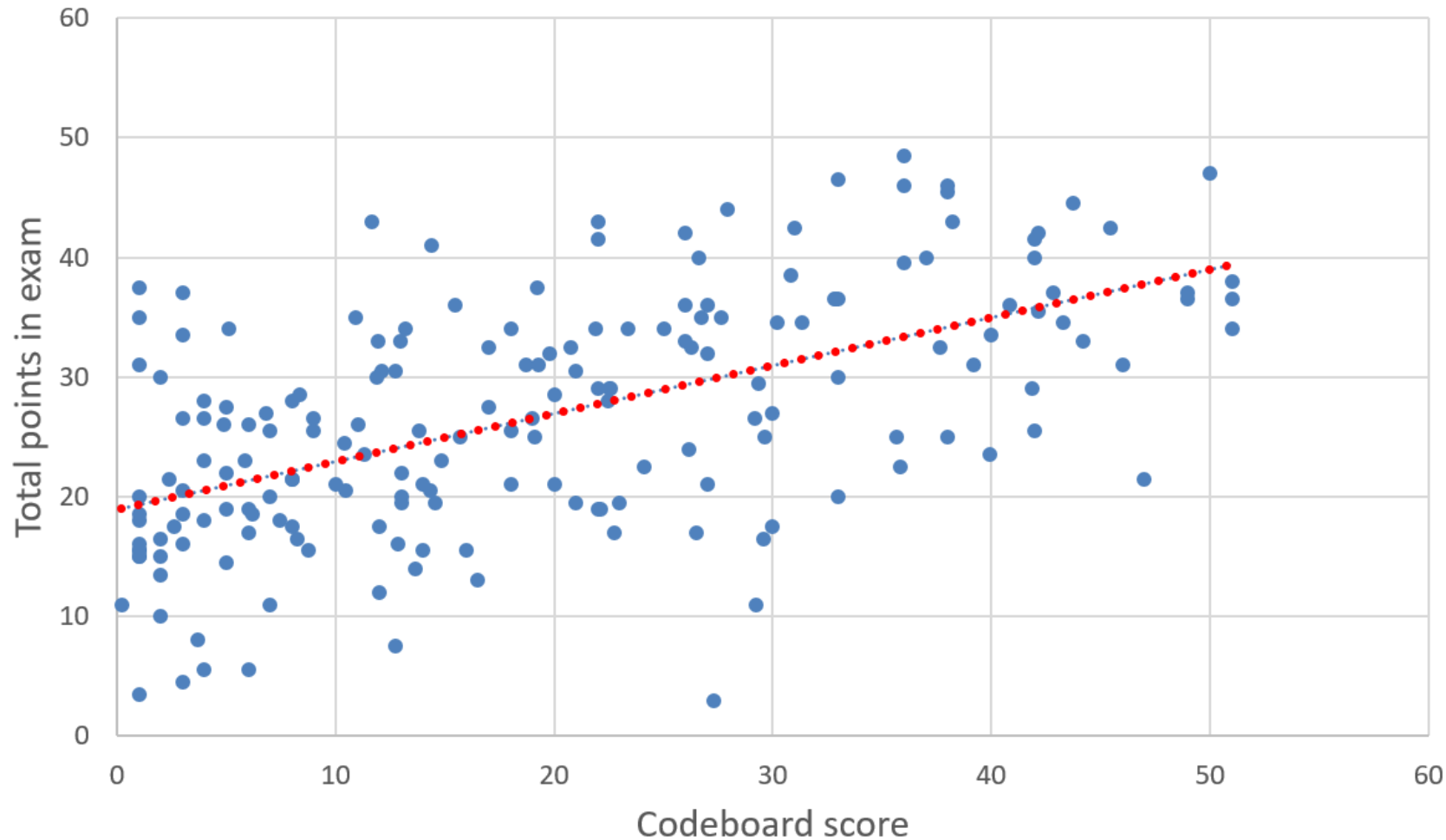
- Es wird wieder spezielle Bonusaufgaben geben (Bonus kann nicht vom letzten Jahr übernommen werden!)
- Um diese freizuschalten, ist das Sammeln von XP in den “normalen” Übungen notwendig

Übungsserie Nr. 0	Earned XP	Submissions	Bonusübung 1
<div style="display: flex; justify-content: space-between;"> Tasks Solutions </div>	3.100 / 3.100		
Hello World	100 ✓	100% new	<div style="text-align: center;">  <p>3.100 XP</p> <p>To unlock, earn additional 14.400 XP to reach the required 17.500 XP in the following exercises (if possible):</p> <ul style="list-style-type: none"> • Übungsserie Nr. 0 ✓ • Übungsserie Nr. 1 (future exercise) • Übungsserie Nr. 2 (future exercise) • Übungsserie Nr. 3 (future exercise) </div>
Das erste Java-Programm	1.000 ✓	100%	
Automatisiertes Testen	1.000 ✓	100%	
Gerichtete Graphen	1.000 ✓	100%	

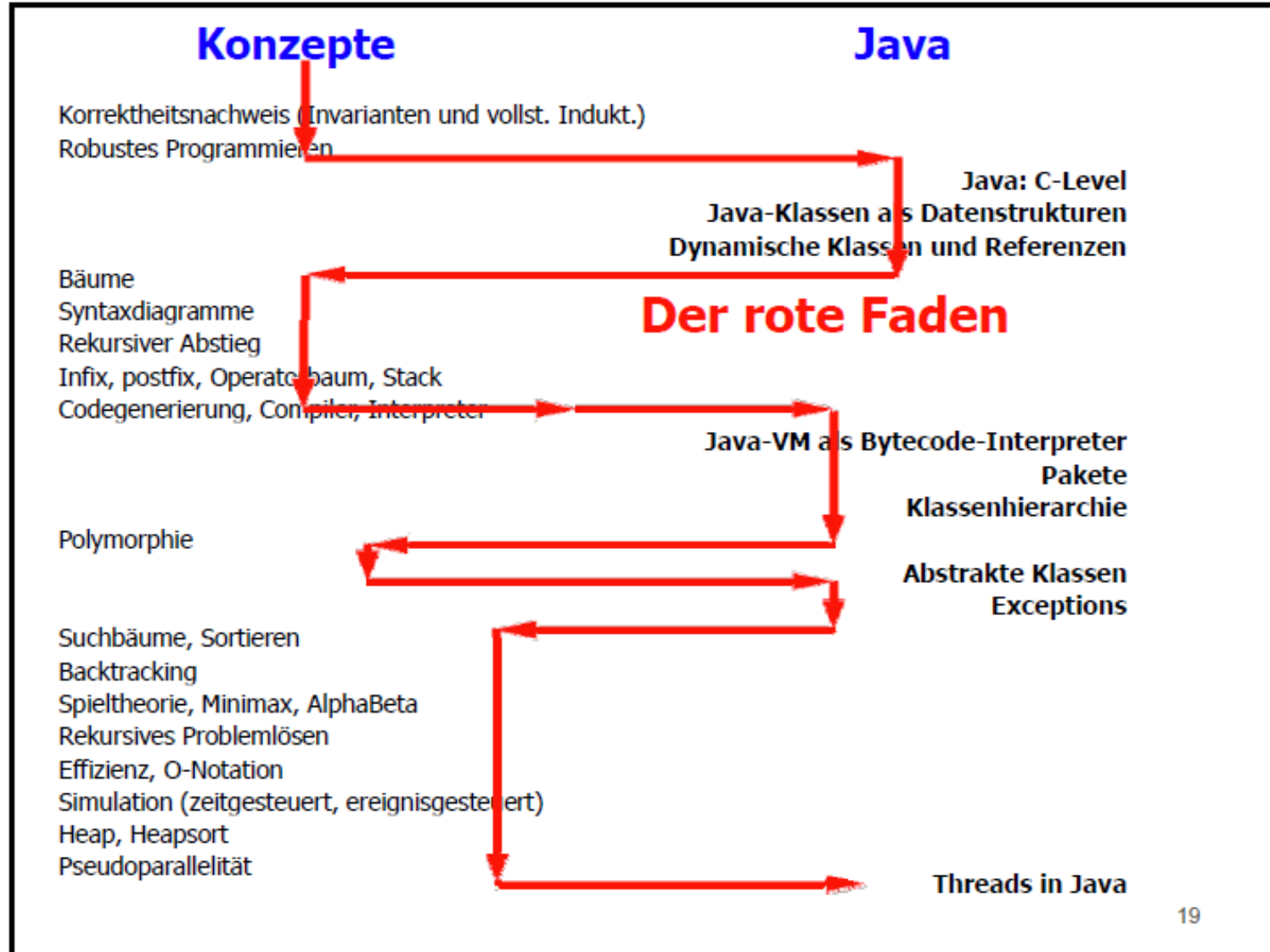
- Die Leistungen aus den Bonusübungen wird proportional in den Bonus von maximal 0.25 Notenpunkten umgerechnet
- Keine Plagiate! Wird automatisch überprüft. Plagiate haben den Verlust des gesamten Bonus aller Beteiligten für das gesamte Semester zur Folge!

Bearbeitung der Übungen führt zu gutem Klausurergebnis?

- Korrelation Codeboard-Score und Punkten in der Klausur: **+0.57**



Konzepte sind wichtig!





(Java ist auch eine Insel)



Warum Java?

- Objektorientiert
- „Einfacher“ als C++
- Umfangreiches Ökosystem: Tools, Bibliotheken, ...
- Virtuelle Maschine: „Compile once – Run everywhere“



“Java is C++ without the guns, knives, and clubs.”
- James Gosling

Werdegang eines Java-Programms

Program.java

Quellcode: Menschenverständlicher Text



`javac Program.java`

Aufruf des **Java-Compilers**

Program.class

Java-Bytecode: Maschinenverständlicher Code



`java Program`

Ausführen mittels **virtueller Maschine**

Plattformunabhängigkeit: Java-Bytecode ist ohne Änderung auf jeder Architektur lauffähig, auf welcher eine Laufzeitumgebung installiert ist.



Hello World!

```
/**  
 * Ein Programm  
 */  
public class HelloWorld {  
  
    public static void main(String[] args) {  
        System.out.println("Hello World");  
    }  
  
}
```

Installation von Java

- Java-Laufzeitumgebung (JRE):
 - Hauptbestandteil ist das Programm *java*
 - Java Virtual Machine (JVM)
 - Standardklassen und weitere Programmbibliotheken
- Java-Entwicklungswerkzeug:
 - Enthält die Programme *java*, *javac* ...
 - Enthält die JRE
- Wir brauchen das JDK! Wir verwenden Java 8



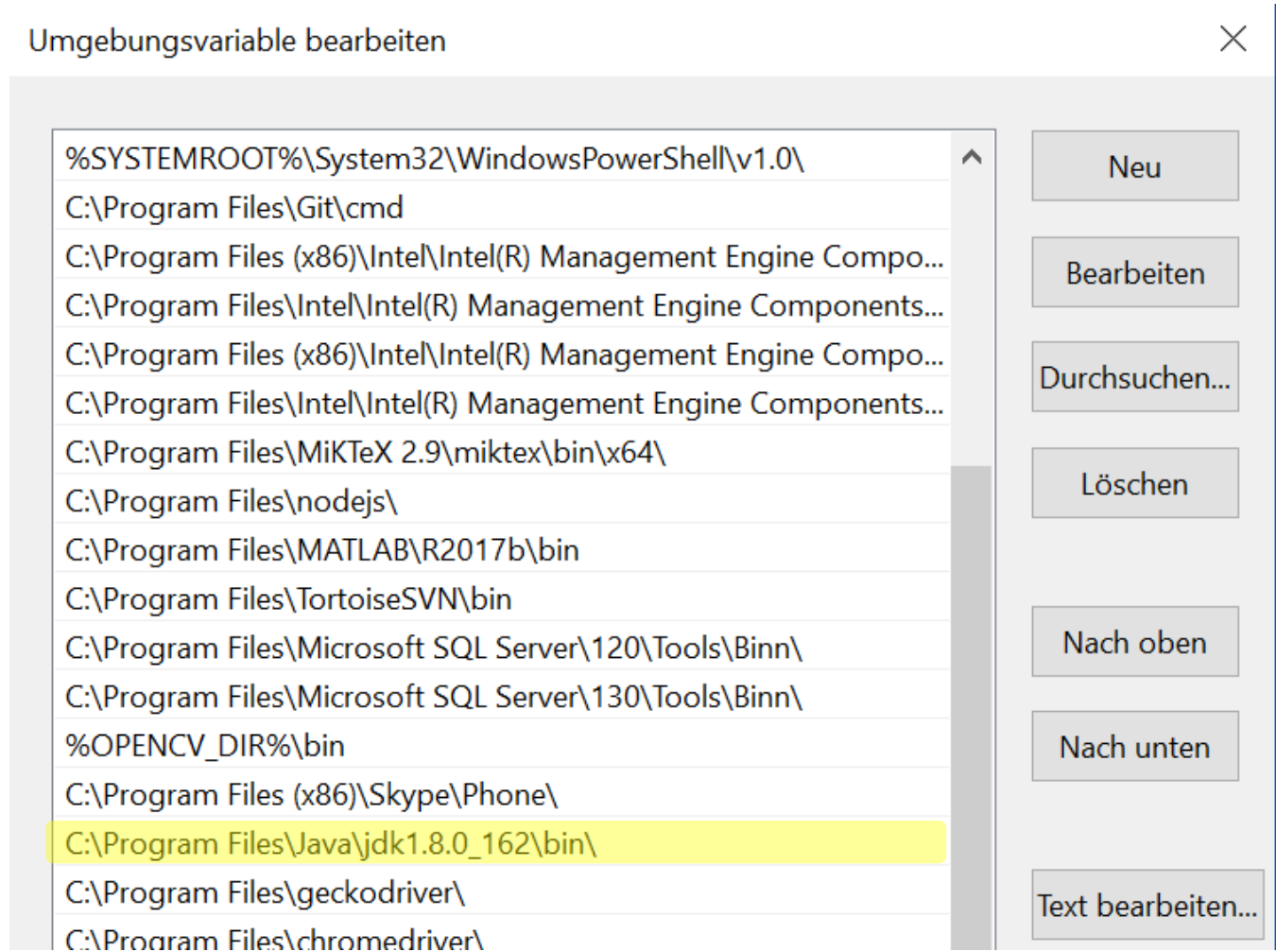
JRE



JDK

Hinweis für Windows

- Nach der Installation setzen der Umgebungsvariablen



Ausführen auf der Konsole

- Aufruf des Compilers

```
C:\Users\Vincent Becker\Desktop\Java_Intro>javac HelloWorld.java
```

- Ausführen des Programms

```
C:\Users\Vincent Becker\Desktop\Java_Intro>java HelloWorld  
Hello World
```

[Demo](#)

Java-Basics

- Primitive Typen können auf dem Stack angelegt werden, ihre Instanzen sind **keine Objekte!**

`boolean`

`byte, char, short, int, long`

`float, double`

- Alle anderen Typen sind Objekttypen

Java-Basics: Überall Objekte!

- Objekt: Instanz einer Klasse
- Zugriff ausschliesslich über Referenzen!
- Erzeugung mit `new`

```
Car c = new Car("Porsche", 300);
```

- Entfernung durch Garbage Collector, **kein delete!**

Java-Basics: Methoden

- Funktionen heissen in Java Methoden
- Jede Methode hat eine eindeutige Signatur

```
public class Car {
```

Zugriffsrechte

Rückgabewert

Name

Parameter

```
    public void driveTo(Location destination, int speed) {  
        System.out.println("Driving to " + destination.toString() + "at a  
        speed of " + speed + "km/h.");  
    }  
}
```

```
}
```

Eclipse (IntelliJIDEA, NetBeans, ...)

- Integrierte Entwicklungsumgebungen bieten viele nützliche Tools
 - Direktes Compilieren und Ausführen
 - Syntaxhighlighting
 - Automatische Checks: Syntax, Typen, ...
 - Autoformat
 - Debuggen
 - Einfaches Testen
 - Refactoring
 - Anzeige von Javadocs
 - ...
- Wir empfehlen die Verwendung von Eclipse, da der Tutor es auf jeden Fall kann
- **WARNUNG:** In der Klausur muss auf Papier programmiert werden!

Installation von Eclipse

- Vorbedingung: JDK installieren
- Eclipse herunterladen: <http://www.eclipse.org/downloads/eclipse-packages/>
 - Eclipse IDE for Java Developers



Eclipse IDE for Java Developers

181 MB

327,484 DOWNLOADS

The essential tools for any Java developer, including a Java IDE, a Git client, XML Editor, Mylyn, Maven and Gradle integration...



Windows

32 bit | 64 bit

- Eclipse ausführen

Ein (bisschen) komplexeres Programm

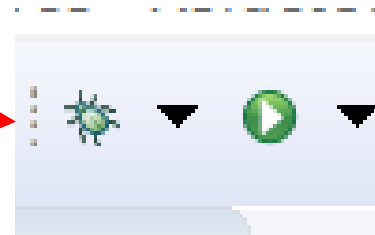
- Aufgabe: Funktion zur Berechnung der Summe aller ungeraden Zahlen bis zur Zahl n



- Achtung: Es gibt viel bessere Lösungen als die naïve, die wir implementieren

Debuggen

- Rechtsklick auf den Rand bei der Zeile an der man anhalten möchte
 - Toggle Breakpoint
- Debugger starten
- Step forward / step into (F6 / F7)



```

10
11 private static int addOdds(int n) {
12     int sum = 0;
13     for (int i = 1; i <= n; i++) {
14         if (i % 2 == 1) {

```

- Toggle Breakpoint Ctrl+Shift+B
- Disable Breakpoint Shift+Double Click
- Go to Annotation Ctrl+1
- Team >
- Add Bookmark...
- Add Task...
- Show Quick Diff Ctrl+Shift+Q
- Show Line Numbers
- Folding >
- Preferences...
- Point Properties... Ctrl+Double Click

```

15 public static int addOdds1(int n) {
16     int sum = 0;
17     for (int i = 1; i <= n; i++) {
18         if (i % 2 == 1) {
19             sum += i;
20         }
21     }
22     return sum;
23 }

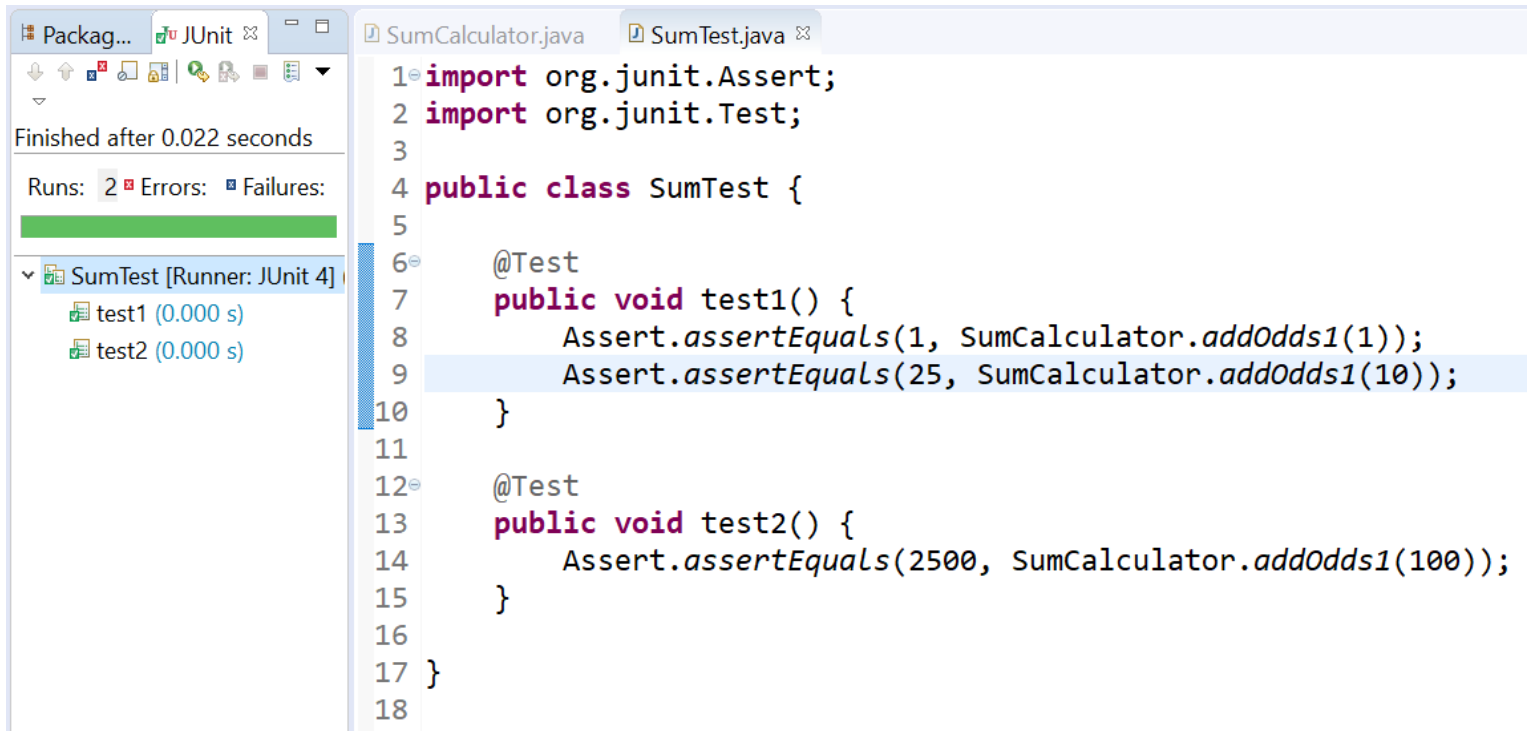
```

Name	Value
n	1
sum	0
i	1

Testen

- Junit Tests: Test einzelner Programm-Komponenten
- Einfach durchführbar in Eclipse
 - New -> JUnit Test Case
 - Run

Demo



The screenshot shows the Eclipse IDE interface. On the left, the 'JUnit' view displays test results for 'SumTest [Runner: JUnit 4]'. It shows two tests: 'test1 (0.000 s)' and 'test2 (0.000 s)', both of which passed. The main editor shows the source code for 'SumTest.java'. The code includes imports for 'org.junit.Assert' and 'org.junit.Test', and defines two test methods: 'test1()' and 'test2()'. The 'test1()' method uses 'Assert.assertEquals' to verify that the sum of 1 and 1 is 2. The 'test2()' method uses 'Assert.assertEquals' to verify that the sum of 100 odds is 2500. The code is as follows:

```
1 import org.junit.Assert;
2 import org.junit.Test;
3
4 public class SumTest {
5
6     @Test
7     public void test1() {
8         Assert.assertEquals(1, SumCalculator.addOdds1(1));
9         Assert.assertEquals(25, SumCalculator.addOdds1(10));
10    }
11
12    @Test
13    public void test2() {
14        Assert.assertEquals(2500, SumCalculator.addOdds1(100));
15    }
16
17 }
18
```

Javadocs

- Strukturierte Kommentare für Java-Code
- Besonders nützlich in Eclipse

```
/**  
 * Calculates the sum of all odd numbers up to n (inclusive).  
 *  
 * @param n The upper bound.  
 * @return The sum of all the odd numbers up to n.  
 */  
public int addOdds1(int n) {  
    ...  
}
```

```
addOdds1(n);
```

```
int SumCalculator.addOdds1(int n)
```

```
Calculates the sum of all odd numbers up to n (inclusive).
```

Parameters:

```
    n The upper bound
```

Returns:

```
    The sum of all odd numbers up to n
```

Press 'F2' for focus

Demo

Übungsblatt 0: Aufgabe 1

- HelloWorld mit Texteditor
- Ausführen auf der Kommandozeile

- HelloWorld in Eclipse
 - Runterladen der Source-Dateien von der Vorlesungswebseite
 - Einbinden in Eclipse

- HelloWorld in CodeExpert
 - Generell empfohlen: Bearbeitung in Eclipse, dann zur Abgabe in CodeExpert kopieren

Übungsblatt 0

- Aufgabe 2
 - Erstes Java-Programm: Signum-Funktion
- Aufgabe 3
 - Automatisiertes Testen mit JUnit4
 - Eclipse
 - CodeExpert
- Aufgabe 4
 - Modellbildung

- Fragen?