Eclipse-Überblick

KW 18, Zentralübung Informatik II

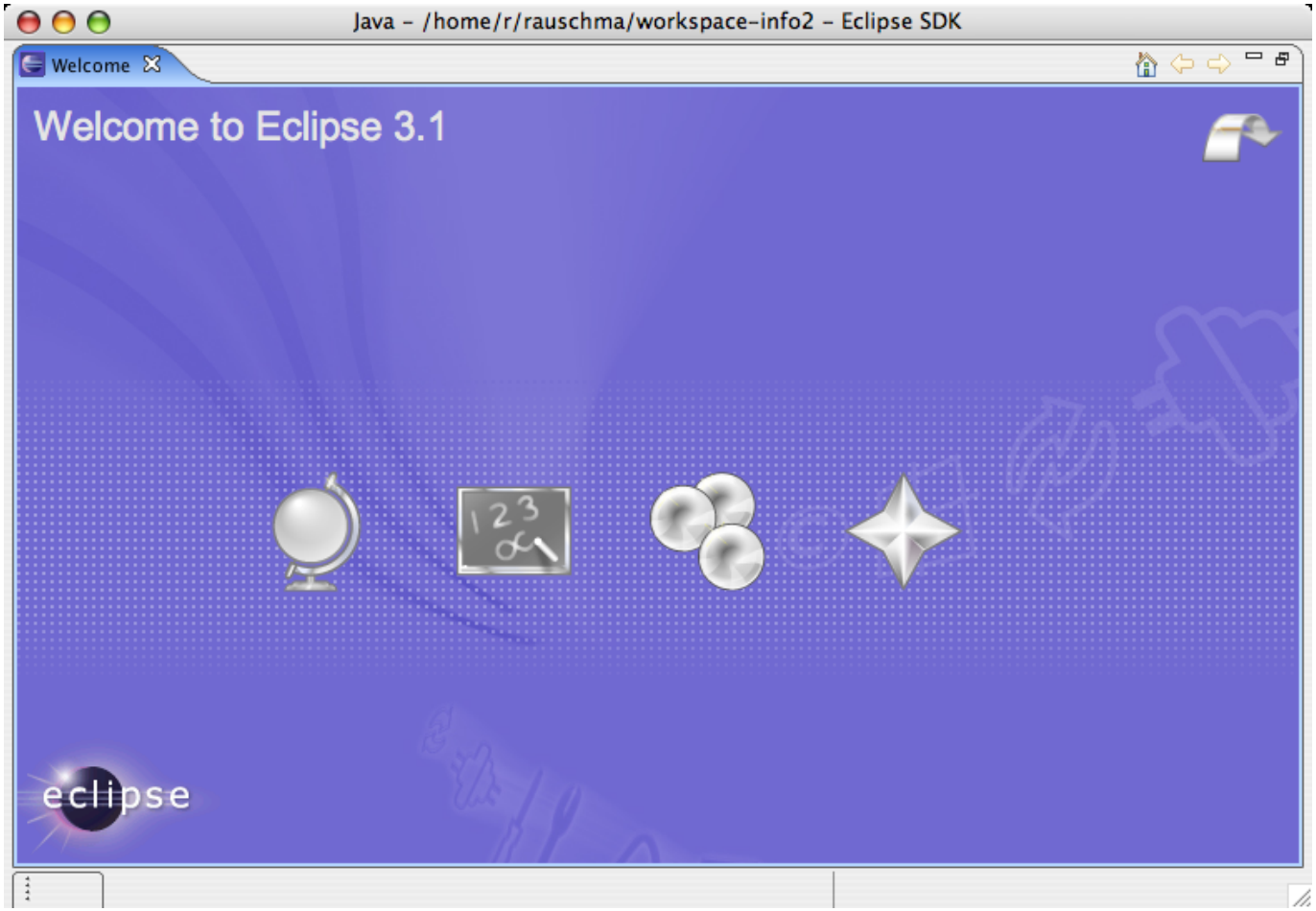
2006-05-02

Java

- Quellcode (.java) wird übersetzt zu Bytecode (.class)
- Bytecode wird interpretiert von der JVM (Java Virtual Machine)
- Momentan: Klassen als Module, die Variablen und Methoden enthalten.
- Bald: Klassen als Erzeuger von $Objekten$.
- Bald: $Packages$ als „Verzeichnishierarchie“ für Klassen.

```
System.out.println()
```

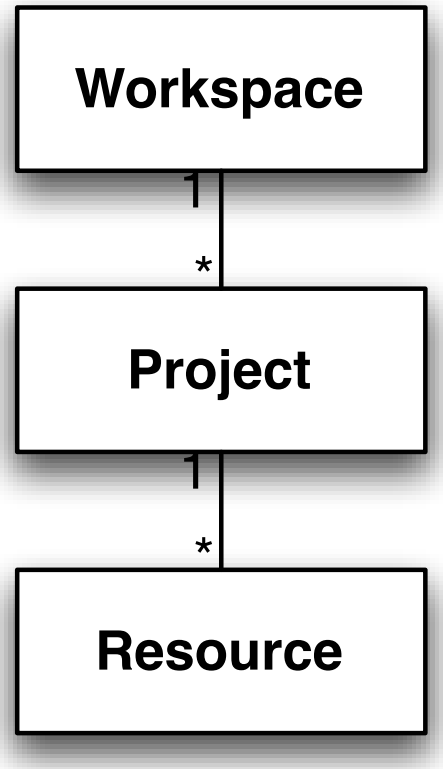
Eclipse


Java - /home/r/rauschma/workspace-info2 - Eclipse SDK
Welcome
Welcome to Eclipse 3.1
eclipse

Warum Eclipse?

- Vereinfachtes Editieren und Übersetzen (Fehlermeldungen, . . .).
- Refactoring
- Navigieren durch den Quellcode
- Verschiedene Sichten auf den Quellcode: Wer ruf alles diese Methode auf? Wer verwendet Klasse X?
- Versionskontrolle

Datenverwaltung



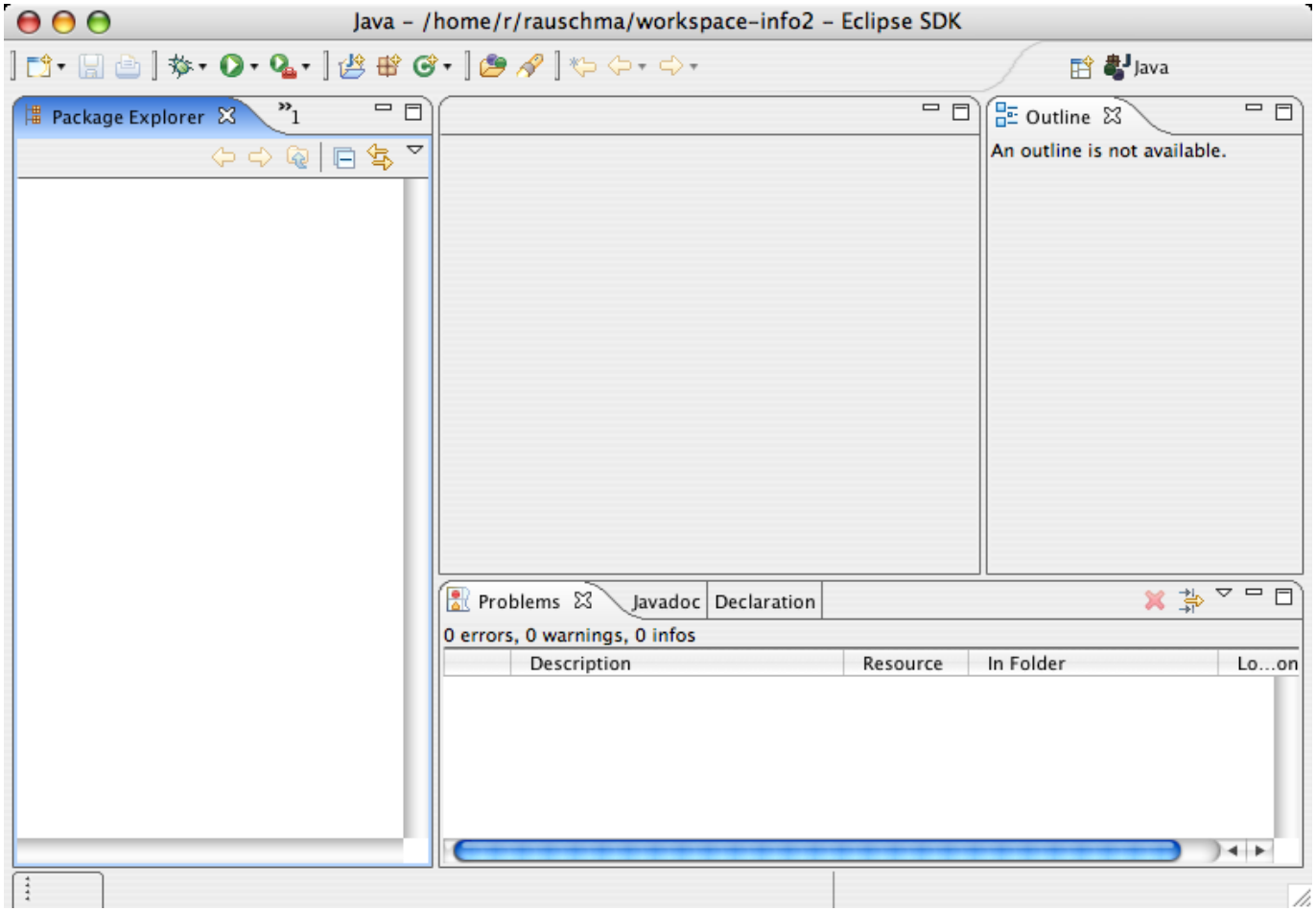
Datenverwaltung

- Workspace
  - Teil des Dateisystems, der von Eclipse verwaltet/überwacht wird
  - Enhält ausserdem Benutzerdaten wie Fensterpositionen etc.
- Project: Der eigentlichen Datencontainer
  - Entweder im Workspace
  - oder als referenz auf externes Verzeichnis.
- Resource: Normalerweise Dateien, aber prinzipiell sind auch andere Datenquellen denkbar z.B. Datenbanken.

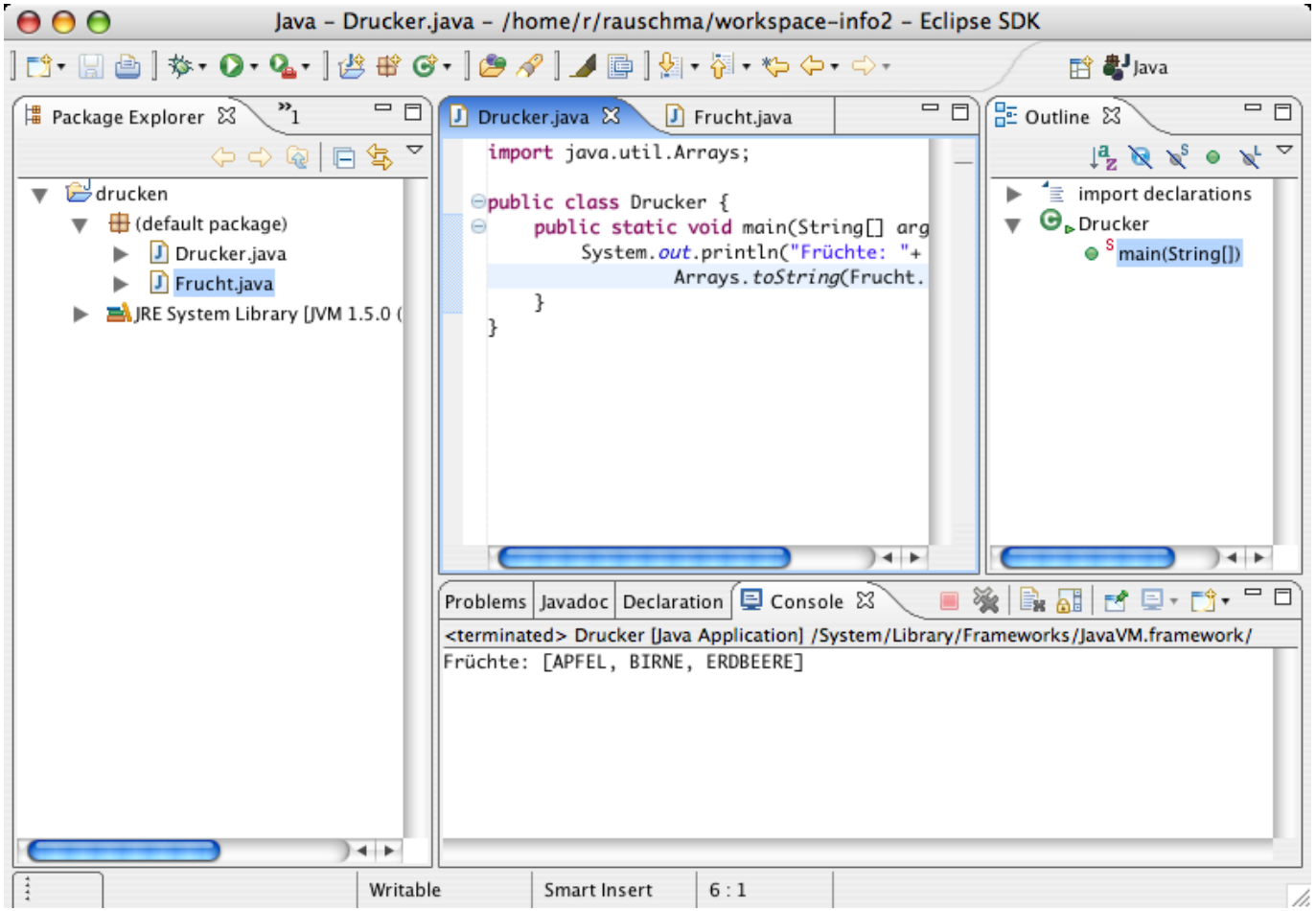
Datenverwaltung

- Man kann mehrere Workspaces einsetzen.
- Man kann Daten, z.B. per Shell, manipulieren, benötigt danach aber einen „Refresh“ (per Kontextmenü), damit Eclipse wieder auf dem aktuellen Stand ist.

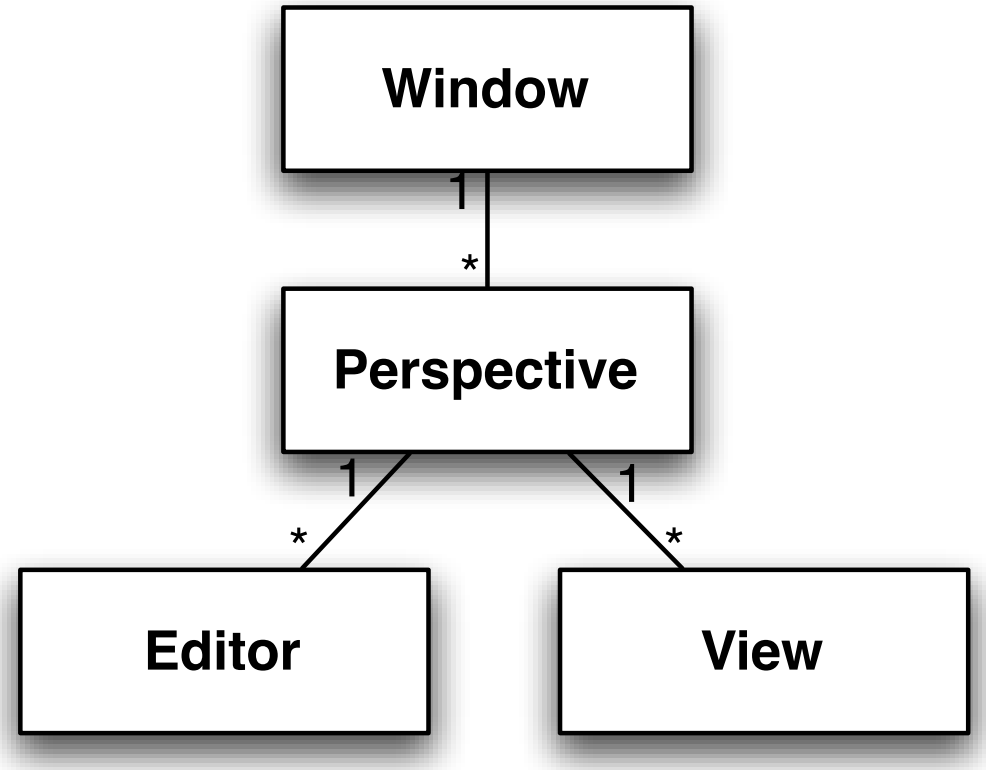
Benutzeroberfläche



Benutzeroberfläche



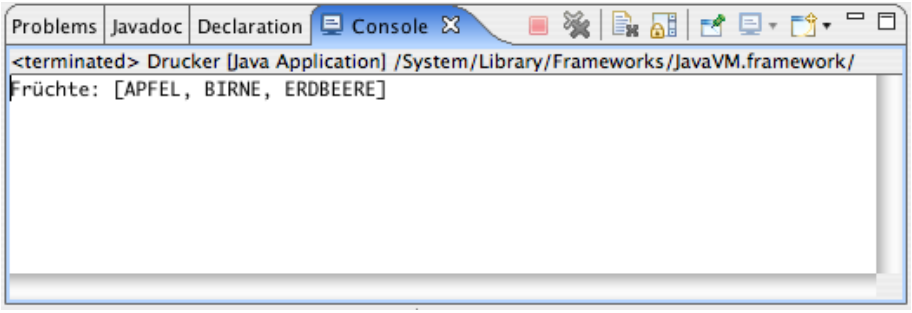
Benutzeroberfläche



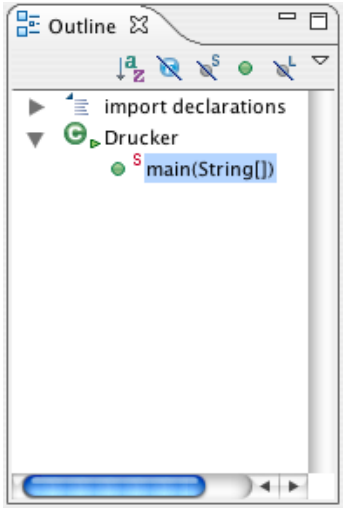
Benutzeroberfläche

- Window: enhält die Perspektiven
- Perspective: eine Sammlung von Views, die einer bestimmten Aufgabe dienen
  - Beispiel: CVS, Java, Debug
- Editor: zum Bearbeiten von Resourcen (z.B. Java-Dateien)
- View:
  - Editor-abhängig: Ein anderer Blick auf den Editor. Beispiel: Outline View.
  - Global: Daten, die alle Editoren betreffen. Beispiel: Package Explorer.

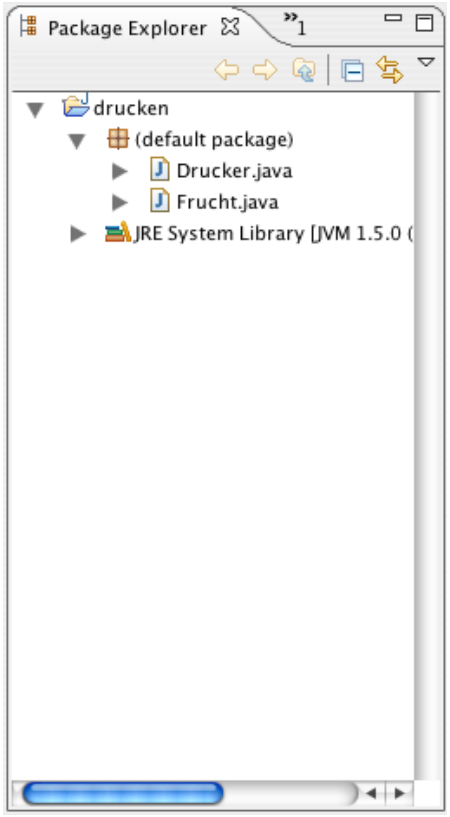
Views: Ein- und Ausgaben



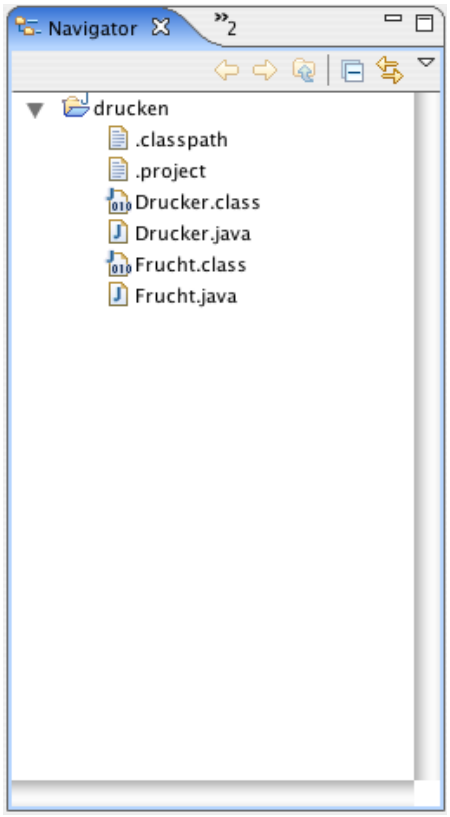
Views: Struktur der bearbeiteten Resource



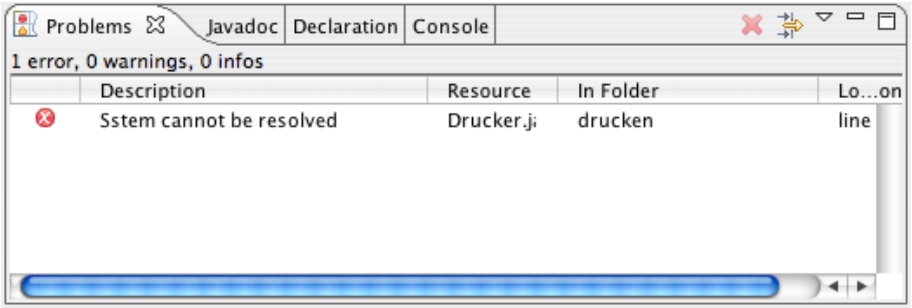
Views: Der Workspace aus Java-Sicht



Views: Der Workspace aus Datei-Sicht



Views: Fehler und Warnungen



Menüs

- File $\rightarrow$ New: Erzeuge Klassen, Projekte etc.
- Source: Formatieren, Quellcode automatisch generieren
- Refactor: Namen ändern etc.
- Navigate: Aufrufhierarchie, gehe zu Typ etc.
- Search: Kann Java-Syntax berücksichtigen (Zeige alle Methoden, deren Name „foo“ enthält).

Menüs

- Project: Projekt-Einstellungen, z. B. Project *Properties* (s. u.).
- Run: Führe die momentan geöffnete Klasse aus (genauer: Die Main-Methode der Übersetzung des momentan offenen Quellcodes).
- Window: Erzeuge Views, Perspektiven. Editiere *Preferences* (s. u.).
- Help: Cheat sheets (Wizard-ähnliche Anleitung zu bestimmten Themen), Tutorials etc.

Einstellungen

Es gibt zwei Arten von Einstellungen:

- Properties (z.B. per Kontextmenü): Projekt- oder Datei-spezifische Einstellungen.
- Preferences (Menü „Window“): Workspace-übergreifende Einstellungen.
  - Java → Installed JREs: JVM einstellen.
  - Java → Compiler → Compiler compliance level: Soll der Compiler Klassendateien passend für Java 5, Java 1.4 etc. erzeugen?

Tipps

- Eclipse-Einführungen: Link im Übungsblatt zu einer deutschen Eclipse-Einführung, Eclipse-Hilfe.
- Abkürzungs-Glossar: Siehe Info-II-Homepage.
- Google is your friend: selbst erforschen, es gibt viel Material im Web.
- die-informatiker.net: Allgemeiner Austausch zu Info-II-Themen. Teilweise antworten auch wir; wir bekommen aber auf jeden Fall Feedback darüber, was (zu) kompliziert ist.
- Tutorien: Da es im ersten Übungsblatt um Eclipse geht, sind die ersten Tutorien eine gute Gelegenheit, Eclipse-Fragen zu stellen.