Software-Qualität: Übung 2
Debugging

Program Slicing
Hypothesen
Reproduzierbarkeit von Fehlern

University of Zurich
Department of Informatics

ifi

Inhalt

Ziele

- Übung 2 erklären

Agenda

- Zielsetzung und Ablauf der Übung
- Aufgaben

Zielsetzung

- Wichtige Debugging-Ansätze anwenden können
  - Program Slicing
  - Erstellen von Fehlerhypothesen
  - Umgang mit Hypothesen über Programmzustände
- Einflussfaktoren auf Reproduzierbarkeit von Fehlern verstehen

Administratives

- Ablauf
  - 7. Mai: Ausgabe der Übung und Einführung
  - 14. Mai: Fragemöglichkeiten (Voranmeldung)
  - 18. Mai, 18:00 CET: Deadline für Abgabe
- Form
  - Individuell, paarweise, oder zu dritt (Ausnahme)
- Infrastruktur
  - Werkzeug: Büroanwendungen und Eclipse
  - Fragen: Uniboard Diskussionsforum und cramer@ifi.unizh.ch
  - Abgabe: cramer@ifi.unizh.ch und fricker@ifi.unizh.ch

Aufgaben

- Aufgabe A:
  Program-Slicing
- Aufgabe B:
  Fehlerhypothesen
- Aufgabe C:
  Reproduktion von Fehlern
- Aufgabe D:
  Hypothesen über Programmzustände

Teil A: Program Slicing

- Gegeben kurzes C-Programm
- Aufgabe
  - a) Set-Use, Use-Set, Set-Set Beziehungen
  - b) Program Dependency Graph (PDG)
  - c) Backward- und Forward-Slice
  - d) Probleme, welche mit PDG entdeckt werden können

Teil A: Set-Use, Use-Set, Set-Set Beziehungen

Programm

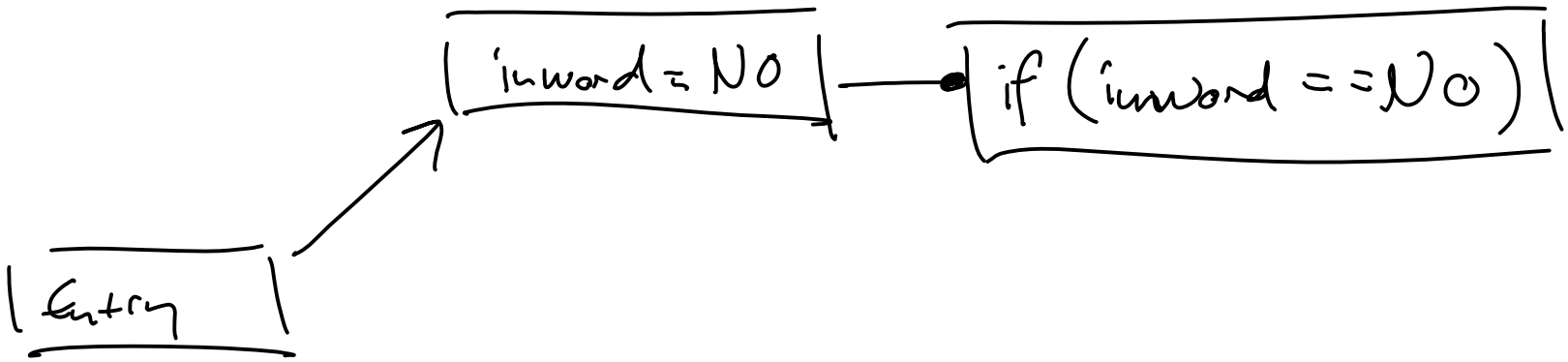
```
06  inword = NO;
...
17      if ( inword == NO )
```

Set-Use Beziehung

- 6 – 17: (inword=NO;) – (if (inword==NO))

- Alle möglichen Kombinationen von Beziehungen erarbeiten
  - jedoch nicht Set-Set-Use, etc.

Teil A: Program Dependency Graph



Kontrollabhängigkeit (Steuerfluss)

Datenabhängigkeit (Datenfluss)

Integrieren Sie alle Beziehungen in einen einzigen Graphen.

Teil A: Statische Slices

Slices

- Programmcode kopieren
- Erreichte Zeilen: stehen lassen
- Nicht erreichte Zeilen: durchstreichen

Ø Verwenden Sie den PDG

Teil B: Fehlerhypothesen

Achtung: Internet-Explorer kann *.jar zu *.zip ändern.
→ einfach umbenennen.

- Gegeben NumberTranslator.jar
  - Input: 21 → Output „twenty one“
- Aufgabe
  - Fehler finden und beschreiben, welcher sind in NumberTranslator.jar eingeschlichen hat
  - Komplette Historie der Diagnosen abgeben
- Diagnose
  - Hypothese:
    Zehnerziffer wird dekrementiert
  - Testfälle (Input → Output):
    12 → 2, 23 → 13, 45 → 35
  - Testresultate (Input → Output):
    12 → 12, 23 → 23, 45 → 45
  - Hypothese soll verworfen werden

Teil C: Reproduktion von Fehlern

- Was sind Einflussfaktoren auf die Reproduzierbarkeit von Fehlern?
  - 2 Beispiele von Faktoren
    Beispiel eines vom Faktor abhängigen Fehlers
    Reproduktion des Fehlers

- Bug Story „Program only works on Wednesday“
  - Faktor: Wochentage (Zeit) und die länge deren Namen
  - Fehler: Benützereingabe wird an allen Wochentagen ausser am Mittwoch falsch interpretiert
  - Reproduktion: Systemzeit auf einen nicht-Mittwoch setzen

Teil D: Hypothesen über Programmzustände

- Gegeben
  - Implementation einer Variante von Quicksort
  - Input, welcher einen Fehler verursacht

- Aufgabe
  - a) Einfachsten Input finden, der den Fehler proviziert
  - b) Hypothese über korrekten Verlauf des Programmzustandes definieren
  - c) Vergleichen des Verlaufs des Ist-Programmzustandes mit der Hypothese
  - d) Defektlokalisierung und Korrekturvorschlag

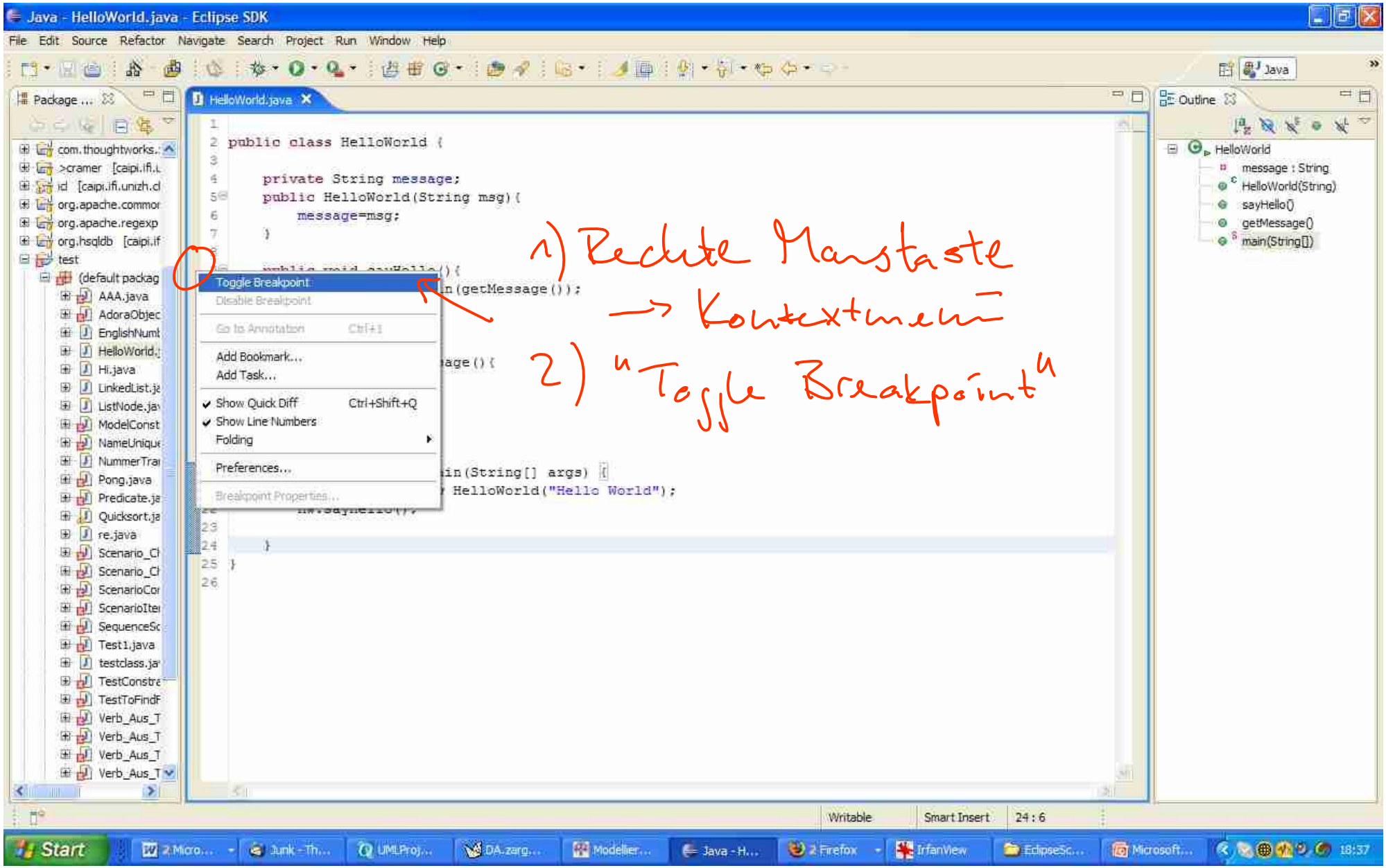
Teil D: Hypothesen über Programmzustände

Programmzustandsbeschreibung

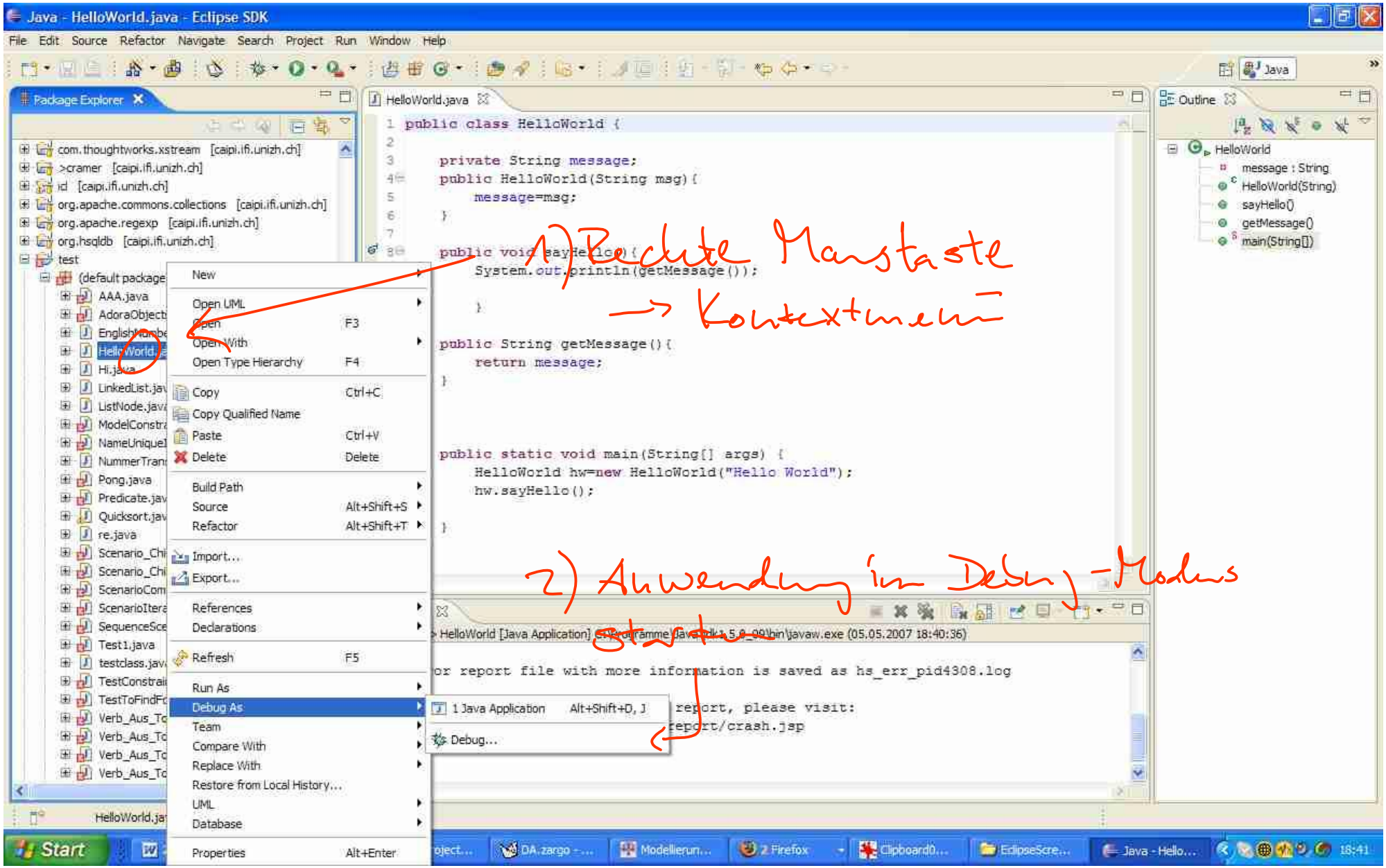
- Debug-Punkt: [Zeilennummer]
- Programmzustant: Werte der Variabeln `start`, `end`, `untergrenze`, `obergrenze`, `vergleichemit`, `elemente`

- Zeigen Sie tabellarisch, wie sich der Programmzustand verändern soll, bzw. dies auch tut → Unterschiede!

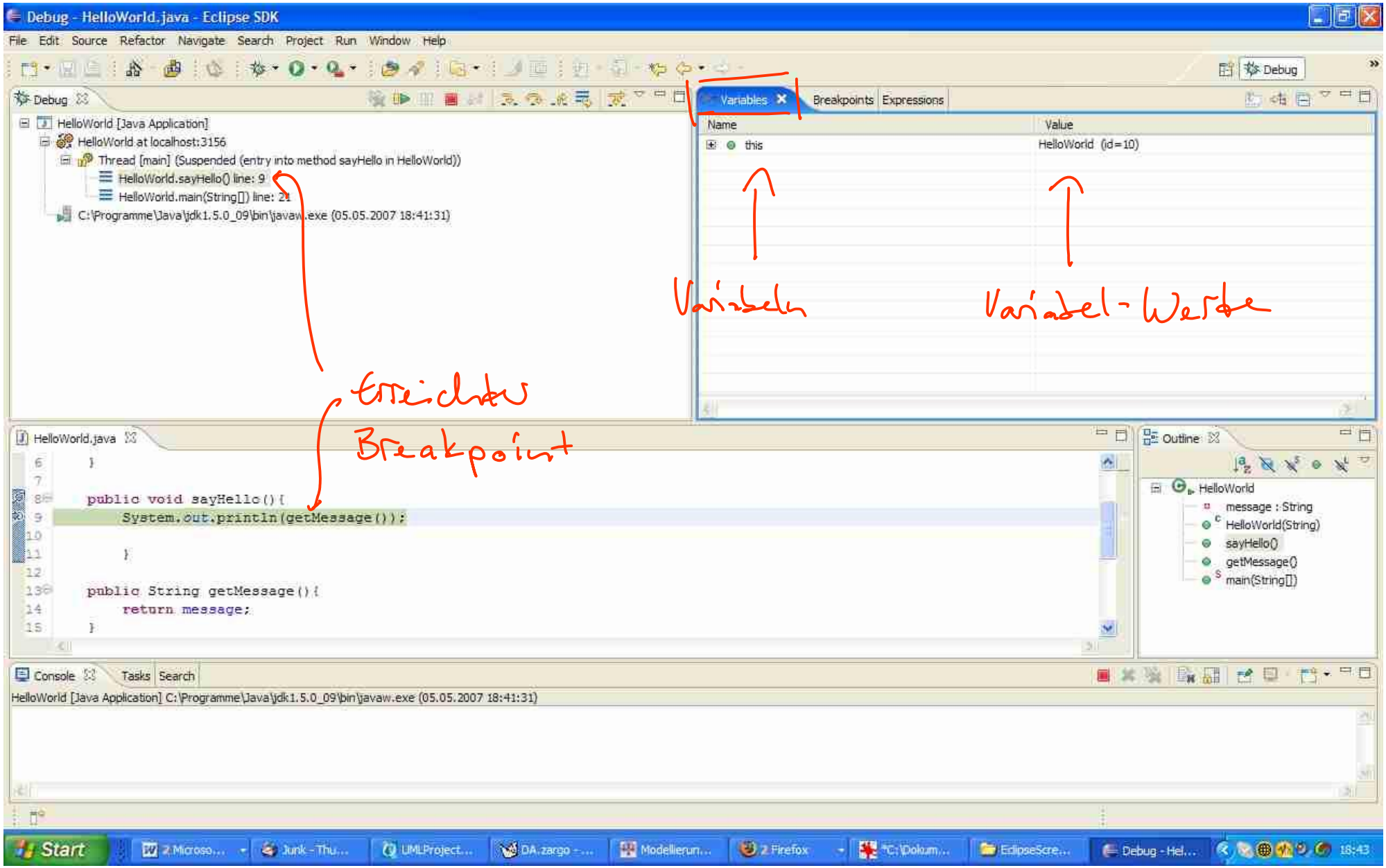
Teil D: Debugging mit Eclipse



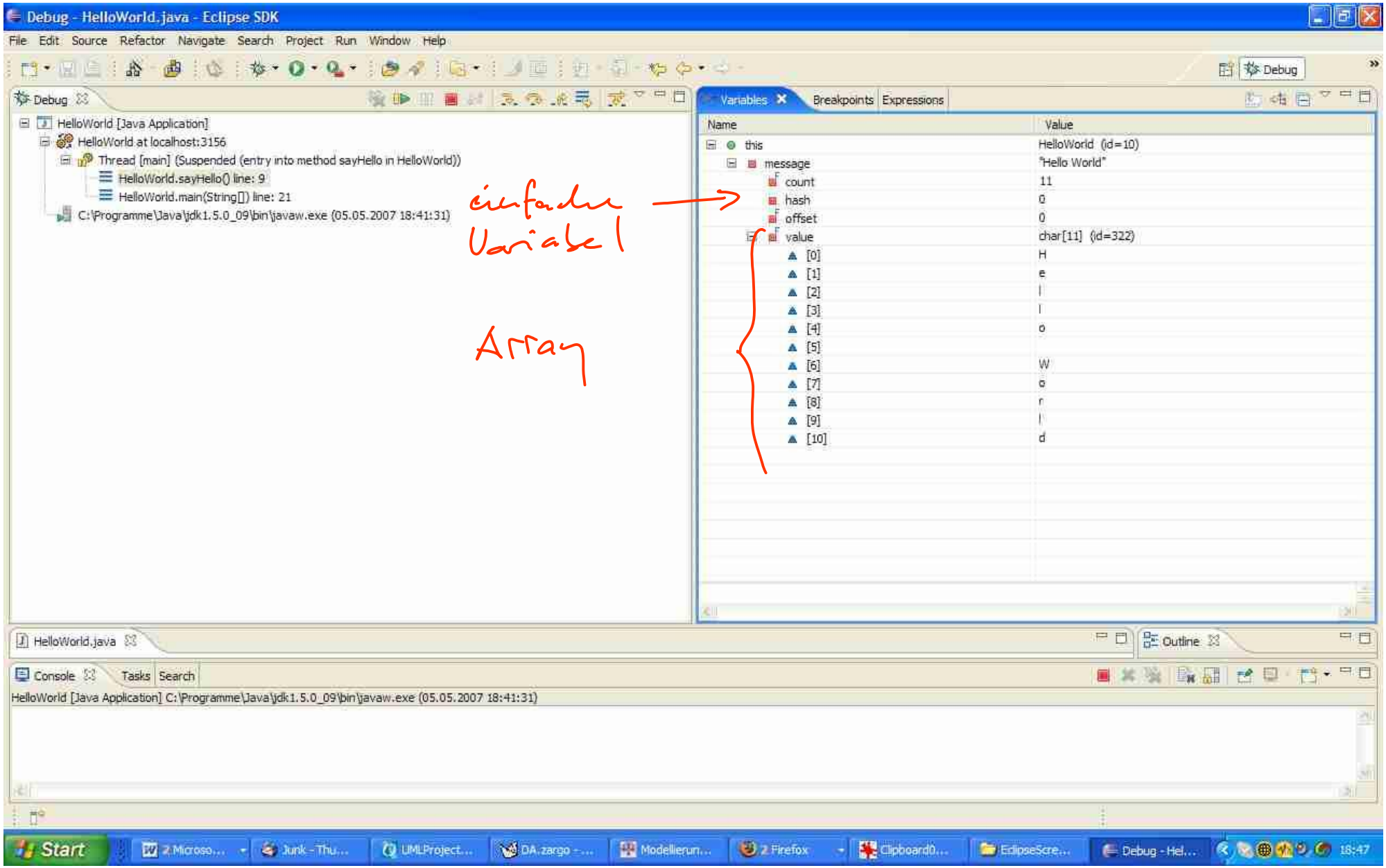
Teil D: Debugging mit Eclipse



Teil D: Debugging mit Eclipse



Teil D: Debugging mit Eclipse



Viel Glück!

- Ablauf
  - 7. Mai: Ausgabe der Übung und Einführung
  - 14. Mai: Fragemöglichkeiten (Voranmeldung)
  - 18. Mai, 18:00 CET: Deadline für Abgabe
- Form
  - Individuell, paarweise, oder zu dritt (Ausnahme)
- Infrastruktur
  - Fragen: Uniboard Diskussionsforum und cramer@ifi.unizh.ch
  - Abgabe: cramer@ifi.unizh.ch und fricker@ifi.unizh.ch
- Gesamtaufwand
  - ca. 10h
  - Zeitaufschreibung machen und Probleme **frühzeitig** melden