

Software-Qualität: Übung 1

Evolutionäres Entwickeln

Versionskontrolle
Automatisiertes Testen
Bugtracking



University of Zurich
Department of Informatics



Inhalt

Ziele

- § Übung 1 erklären
- § Überblick über
Werkzeuge geben
 - § JUnit
 - § CVS
 - § Bugzilla

Agenda

- § Zielsetzung, Struktur und
Ablauf der Übung 1
- § Aufgabe 1:
Regressionstests mit
JUnit
- § Aufgabe 2:
Kollaborative Entwicklung
mit JUnit, CVS, und
Bugzilla

Zielsetzung

- § Die Rolle von Regressionstests für die Minimierung von Fehlern bei ändernder Software verstehen
- § Die Rolle von Regressionstests, Konfigurationsmanagement, und Bugtracking für die gemeinschaftliche Softwareentwicklung verstehen
- § Verbreitete Werkzeuge verwenden können
 - § Eclipse mit JUnit, CVS und Bugzilla

§ Teil A: Regressionstests

§ A.1: Anwendung im Test-First Ansatz entwickeln

§ A.2: Probleme verstehen, welche bei Änderungen entstehen

§ A.3: Probleme eliminieren

§ Teil B: Kollaborative Entwicklung

§ Aufträge über Bugtracking-System entgegennehmen und nachverfolgen

§ Entwicklung mit kontinuierlicher Integration

§ Test-First Entwicklungsansatz weiter vertiefen

Administratives

§ Ablauf

- § 2. April: Ausgabe der Übung und Einführung
- § 10. April: Fragemöglichkeiten
- § 11. April: Aufträge and Partner in Bugzilla eingetragen (Übung 2)
- § 17. April: Fragemöglichkeiten
- § 20. April, 18:00 CET: Deadline für Abgabe

§ Form

- § Übung A: Individuell
- § Übung B: in Gruppen von 2 Personen
 - § 1 Person für Konto.java, 1 Person für Verwaltung.java

§ Infrastruktur

- § Fragen: Uniboard Diskussionsforum und cramer@ifi.unizh.ch
- § Abgabe: cramer@ifi.unizh.ch und fricker@ifi.unizh.ch

Inhalt

Ziele

- § Übung 1 erklären
- § Überblick über
Werkzeuge geben
 - § JUnit
 - § CVS
 - § Bugzilla

Agenda

- § Zielsetzung, Struktur und
Ablauf der Übung 1
- § Aufgabe 1:
Regressionstests mit
JUnit
- § Aufgabe 2:
Kollaborative Entwicklung
mit JUnit, CVS, und
Bugzilla

Übung A: Regressionstests

A.1

§ MathematikBibliothek

§ public boolean isNumber(Object entry)

§ public int multiplyByHundred(int number)

§ public Vector getMultiplyByHundred(Vector entries)

§ CalculatorTestCase

§ public void testIsNumber()

§ public void testMultiplyByHundred()

§ public void testGetMultiplyByHundred()

§ public static Test suite()

Übung A: Regressionstests

A.2

§ MathematikBibliothek2

§ public boolean isNumber(Object entry)

§ public boolean isPrime(int number)

§ public Vector getPrimeNumbers(Vector entries)

§ CalculatorTestCase2

§ public void testIsNumber()

§ public void testIsPrime()

§ public void testGetPrimeNumbers()

§ public static Test suite()

Übung A: Regressionstests

A.3

§ MathematikBibliothek3

§ integration von MathematikBibliothek und MathematikBibliothek2

§ CalculatorTestCase

§ integration von CalculatorTestCase und CalculatorTestCase2

Übung A: Vorgehen

Vorbereitung

- § Eclipse installieren
- § Skeleton von CVS Repository laden
- § Projekt von CVS entkoppeln
Rechte Maustaste > Team > Disconnect

Implementierung

- § Methodensignaturen schreiben
- § Für jede Methode
 - § Tests schreiben
 - § Methode implementieren
 - § Testen

§ Fragen beantworten

Rückgabe

- § per e-Mail (ZIP-Datei)

File > Import > CVS > Project from CVS

Checkout from CVS

Enter Repository Location Information

Define the location and protocol required to connect with an existing CVS repository.

Location

Host: 130.60.157.149

Repository path: /swq

Authentication

User: SwQ

Password: ●●●

Connection

Connection type: pserver

Use default port

Use port:

Save password

⚠ Saved passwords are stored on your computer in a file that is difficult, but not impossible, for an intruder to read.

? < Back Next > Finish Cancel

Übung A: JUnit Tests

```
package tests;

import src.Calculator;
import junit.framework.Test;
import junit.framework.TestCase;
import junit.framework.TestSuite;

public class CalculatorTestSuite extends TestCase {

    //hier werden die Testfälle geschrieben.
    private Calculator calculator;

    public void testInstantiateCalculator() {}

    public void testIsNumber() {

        // initialize
        calculator = new Calculator();

        // test sensible inputs and outputs
        assertTrue(...);
        assertFalse(...);
        assert...(...);
        ...;
    }

    public void testMultiplyByHundred() {}

    public void testGetMultiplyByHundred() {}

    public static Test suite() {
        TestSuite suite = new TestSuite(CalculatorTestSuite.class);
        return suite;
    }
}
```

Übung A: JUnit Tests

The image shows two parts of an IDE interface. On the left, a project tree shows a package named 'tests' containing 'CalculatorTestSuite.java'. A context menu is open over this file, with 'Run As' selected, which has opened a sub-menu showing '1 JUnit Test' and 'Run...'. On the right, a 'JUnit' runner window is displayed. It shows the test suite 'tests.CalculatorTestSuite' with four tests: 'testInstantiateCalculator', 'testIsNumber', 'testIsPrime', and 'testGetPrime'. The runner indicates it finished after 0.032 seconds with 4/4 runs, 0 errors, and 0 failures. A green progress bar is visible at the top of the runner window.

Inhalt

Ziele

- § Übung 1 erklären
- § Überblick über
Werkzeuge geben
 - § JUnit
 - § CVS
 - § Bugzilla

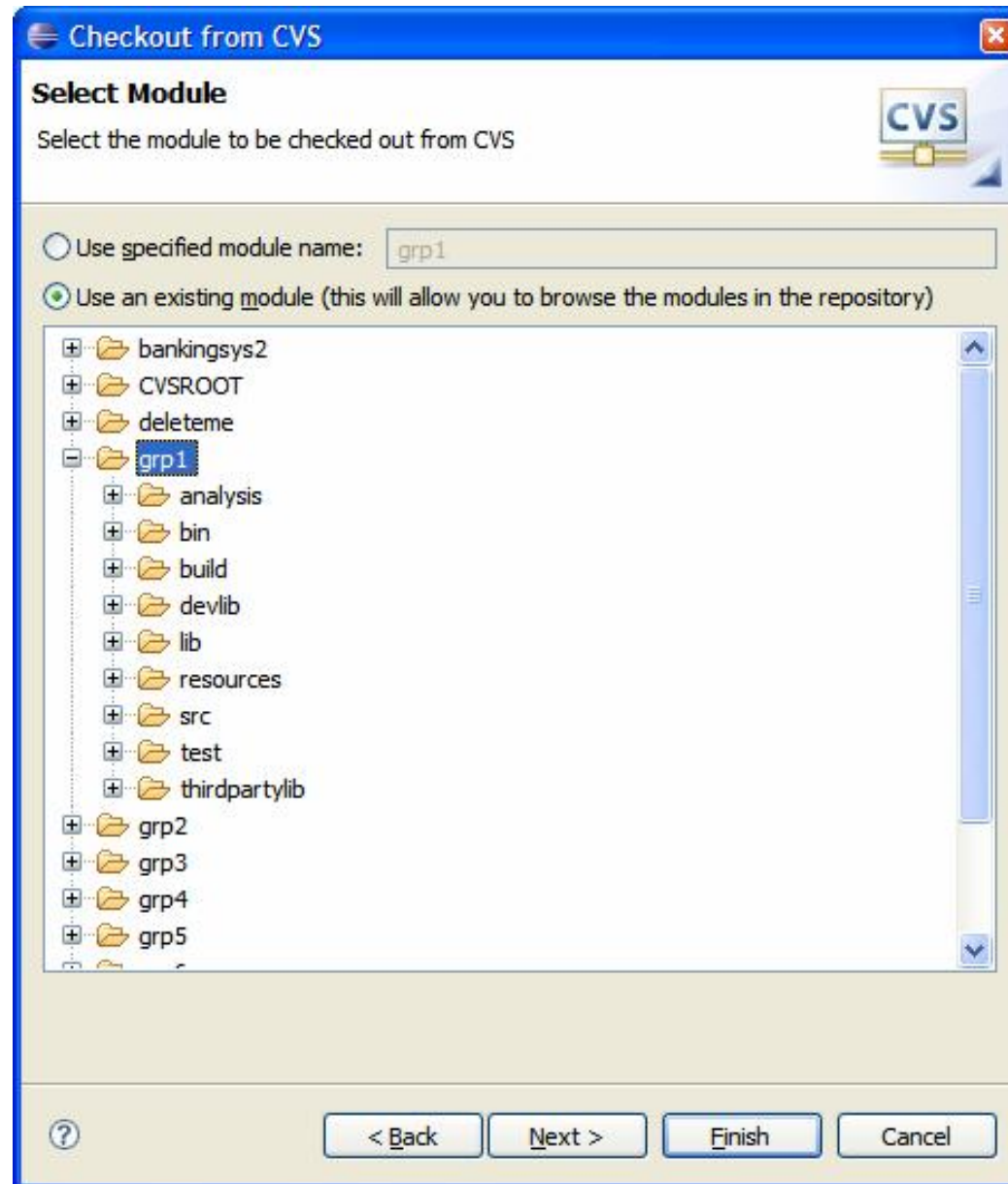
Agenda

- § Zielsetzung, Struktur und
Ablauf der Übung 1
- § Aufgabe 1:
Regressionstests mit
JUnit
- § Aufgabe 2:
Kollaborative Entwicklung
mit JUnit, CVS, und
Bugzilla

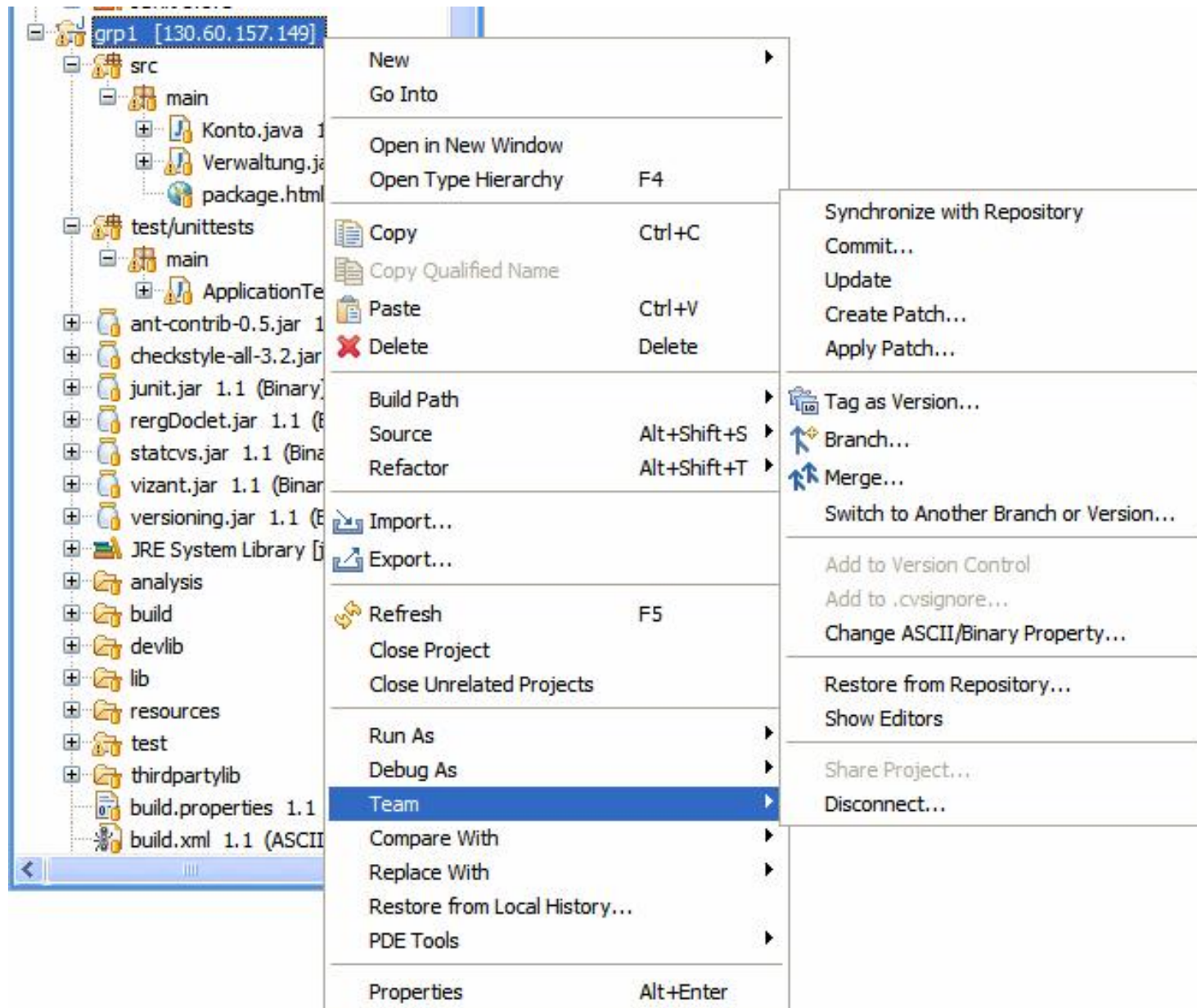
Übung B: Kollaborative Entwicklung

- § Projekt „grpX“ von CVS herunterladen
 - § (X=Gruppennummer)
- § Bugzilla Account eröffnen
 - § <http://130.60.157.149>
 - § Bugzilla Produkt „grpX“ öffnen, um zu den Änderungswünschen zu gelangen
- § Für jeden Änderungswunsch
 - § JUnit Test
 - § Implementierung
 - § Integration in CVS
 - § in Bugzilla Quittieren
- § 4 weitere Änderungswünsche für Ihren Partner in Bugzilla eintragen
- § Änderungswünsche von Ihrem Partner implementieren

Übung B: CVS Operationen



Übung B: CVS Operationen



Übung B: Bugzilla Operationen

Bugzilla

[Bugzilla](#) Version 2.22.2

Bugzilla Main Page

This is where we put in lots of nifty words explaining all about Bugzilla.

But it all boils down to a choice of:

[Search existing bug reports](#)

[Enter a new bug report](#)

[Summary reports and charts](#)

[Change password or user preferences](#)

[Log out fricker@ifi.unizh.ch](#)

[Add to Sidebar](#) (requires a Mozilla browser like Mozilla Firefox)



Enter a bug # or some search terms:

 [\[Help\]](#)

Actions: [Home](#) | [New](#) | [Search](#) | |

[Reports](#) | [My Requests](#) | [My Votes](#) |

[Log out fricker@ifi.unizh.ch](#)

Edit: [Prefs](#)

Saved Searches: [My Bugs](#)

bugs to the new saved search:

Übung B: Bugzilla Operationen

Bugzilla

Bugzilla Version 2.22.2

[Find a Specific Bug](#) [Advanced Search](#)

Find a specific bug by entering words that describe it. Bugzilla will search bug descriptions and comments for those words and return a list of matching bugs sorted by relevance.

For example, if the bug you are looking for is a browser crash when you go to a secure web site with an embedded Flash animation, you might search for "crash secure SSL flash".

Status:

Product:

Words:

Actions: [Home](#) | [New](#) | [Search](#) | | [Reports](#) | [My Requests](#) | [My Votes](#) | [Log out](#) fricker@ifi.unizh.ch

Edit: [Prefs](#)

Saved Searches: [My Bugs](#)

bugs to the new saved search:

Bugzilla

Bugzilla Version 2.22.2

Bug List

Sun Apr 1 2007 21:43:26

Bugzilla would like to put a random quip here, but no one has entered any.

<u>ID</u>	<u>Sev</u>	<u>Pri</u>	<u>OS</u>	<u>Assignee</u>	<u>Status</u>	<u>Resolution</u>	<u>Summary</u>
2	enh	P1	All	cramer@ifi.unizh.ch	NEW		Logging-Mechanismus
3	enh	P1	All	cramer@ifi.unizh.ch	NEW		Konto mit Anfangskontostand
4	enh	P1	Wind	cramer@ifi.unizh.ch	NEW		Speicherung Konten
5	blo	P1	All	cramer@ifi.unizh.ch	NEW		Attribute
6	blo	P1	All	cramer@ifi.unizh.ch	NEW		Konto Limite

5 bugs found.

[CSV](#) | [Feed](#) | [iCalendar](#) | [Change Columns](#) | [Edit Search](#) as

[Change Several Bugs at Once](#) |

Actions: [Home](#) | [New](#) | [Search](#) | | [Reports](#) | [My Requests](#) | [My Votes](#) | [Log out](#) fricker@ifi.unizh.ch

Edit: [Prefs](#)

Saved Searches: [My Bugs](#)

bugs to the new saved search:

Übung B: Bugzilla Operationen

Bugzilla

Bugzilla Version 2.22.2

Bugzilla Bug 2 Logging-Mechanismus Last modified: 2007-04-01 17:23:22

Bug List: (1 of 5) [First](#) [Last](#) [Prev](#) [Next](#) [Show last search results](#) [Search page](#) [Enter new bug](#)

Bug#: 2

Product:

Component:

Status: NEW

Resolution:

Hardware:

OS:

Version:

Priority:

Reporter: [Christina Cramer](#)
<cramer@ifi.unizh.ch>

Add CC:

CC:

Leave as **NEW**

Accept bug (change status to **ASSIGNED**)

Resolve bug, changing **resolution** to

Resolve bug, mark it as duplicate of bug #

Reassign bug to

Reassign bug to default assignee of selected component

Übung B: Bugzilla Operationen

Bugzilla

Bugzilla Version 2.22.2

Enter Bug: grp2

Before reporting a bug, please read the [bug writing guidelines](#), please look at the list of [most frequently reported bugs](#), and please [search](#) for the bug.

Reporter: fricker@fi.unizh.ch

Version: unspecified

Product: grp2

Component: general
Konto
Verwaltung

Platform: PC

Priority: P1

OS: Windows

Severity: blocker

Initial State: NEW

Assign To:

Cc:

URL: http://

Summary:

Description:

Depends on:

Blocks:

Commit

Remember values as bookmarkable template

Viel Glück!

§ Ablauf

- § 2. April: Ausgabe der Übung und Einführung
- § 10. April: Fragemöglichkeiten
- § 11. April: Aufträge and Partner in Bugzilla eingetragen (Übung 2)
- § 17. April: Fragemöglichkeiten
- § 20. April, 18:00 CET: Deadline für Abgabe

§ Form

- § Übung A: Individuell
- § Übung B: in Gruppen von 2 Personen
 - § 1 Person für Konto.java, 1 Person für Verwaltung.java

§ Infrastruktur

- § Fragen: Uniboard Diskussionsforum und cramer@ifi.unizh.ch
- § Abgabe: cramer@ifi.unizh.ch und fricker@ifi.unizh.ch