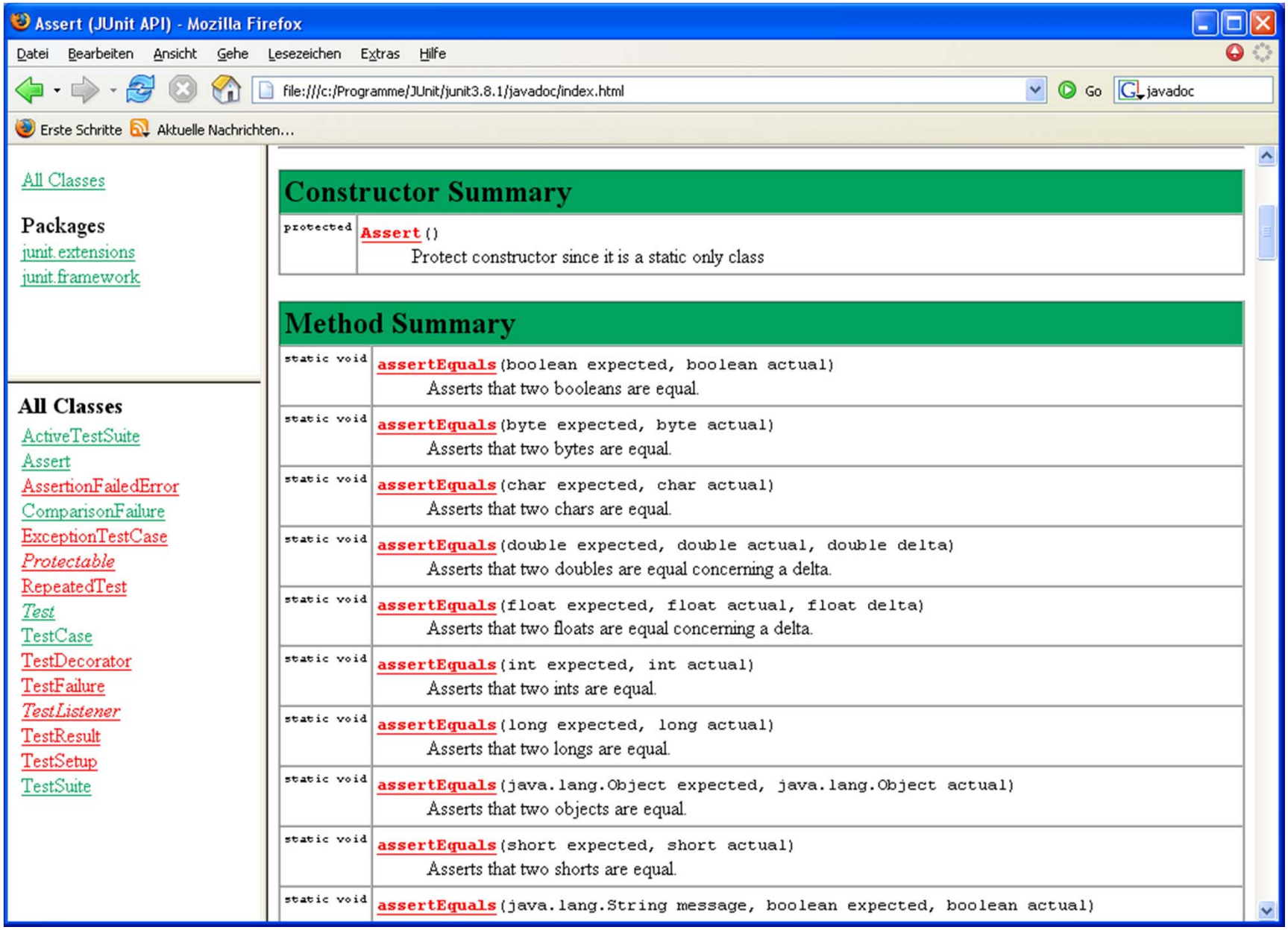
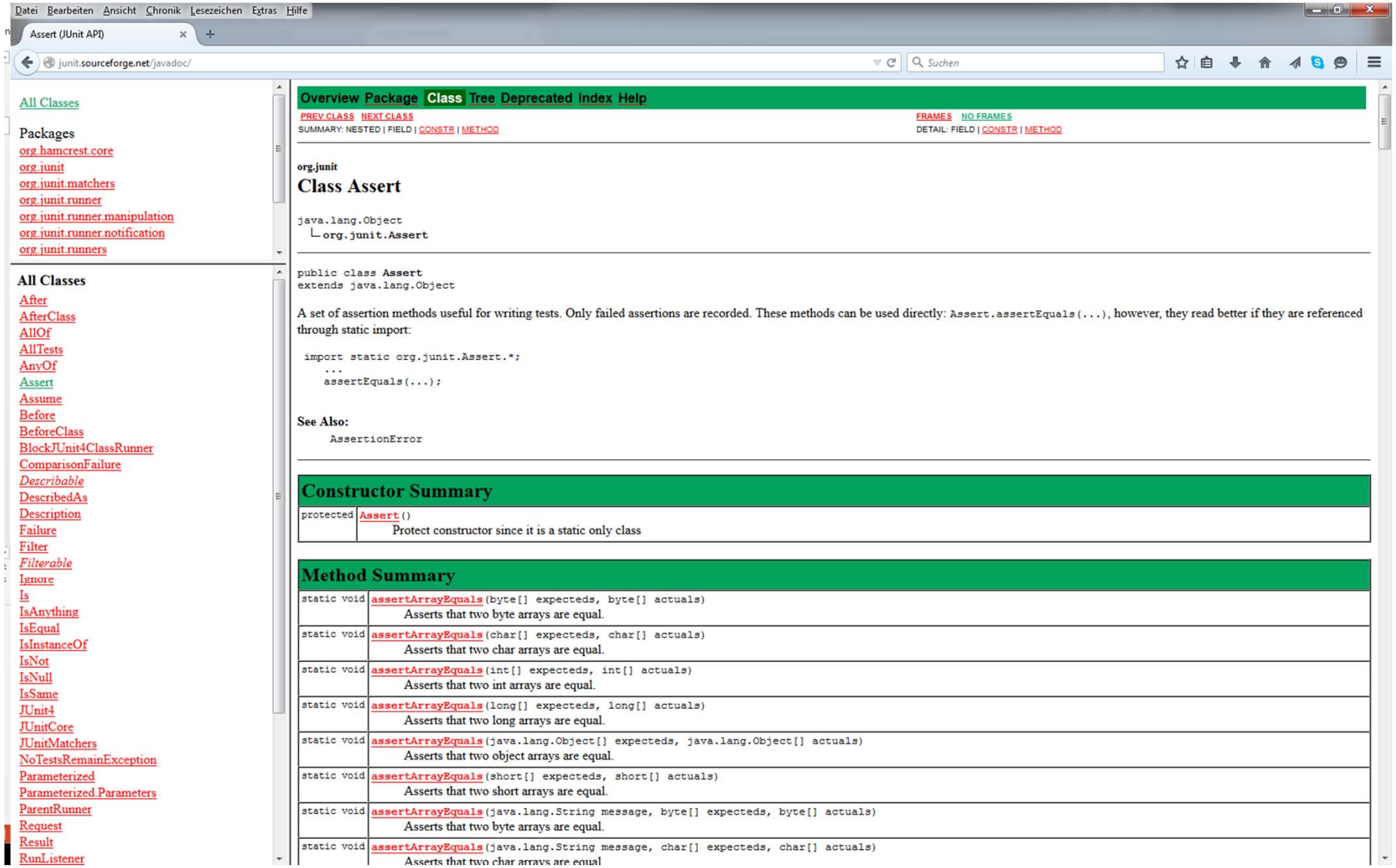
# Musterlösung U04

## Testen mit JUnit
## Softwaretechnologie SS 2016

# Schreiben von Unit-Tests mit JUnit 3.8.1

1. Testfälle ausdenken / Testfalltabellen erstellen
2. Testfälle nach gemeinsamen Fixtures in Klassen gruppieren
3. i. Allg. pro Testklasse eine „Halterung“ (Fixture)
   - Methode **`setUp()`**
   - Methode **`tearDown()`**
   - pro Testfall eine Methode (mit **`„test“`** beginnend) schreiben
4. Mengen von Testfällen (Testklassen) in Testsuiten zusammenfassen und einem Testrunner übergeben
   - **`junit.swingui.TestRunner`**
   - **`junit.textui.TestRunner`**


Assert (JUnit API) - Mozilla Firefox
Datei Bearbeiten Ansicht Gehe Lesezeichen Extras Hilfe
file:///c:/Programme/JUnit/junit3.8.1/javadoc/index.html
Go
javadoc
Erste Schritte
Aktuelle Nachrichten...
All Classes
Packages
junit.extensions
junit.framework
All Classes
ActiveTestSuite
Assert
AssertionFailedError
ComparisonFailure
ExceptionTestCase
Protectable
RepeatedTest
Test
TestCase
TestDecorator
TestFailure
TestListener
TestResult
TestSetup
TestSuite
Constructor Summary
protected Assert()
Protect constructor since it is a static only class
Method Summary
static void assertEquals(boolean expected, boolean actual)
Asserts that two booleans are equal.
static void assertEquals(byte expected, byte actual)
Asserts that two bytes are equal.
static void assertEquals(char expected, char actual)
Asserts that two chars are equal.
static void assertEquals(double expected, double actual, double delta)
Asserts that two doubles are equal concerning a delta.
static void assertEquals(float expected, float actual, float delta)
Asserts that two floats are equal concerning a delta.
static void assertEquals(int expected, int actual)
Asserts that two ints are equal.
static void assertEquals(long expected, long actual)
Asserts that two longs are equal.
static void assertEquals(java.lang.Object expected, java.lang.Object actual)
Asserts that two objects are equal.
static void assertEquals(short expected, short actual)
Asserts that two shorts are equal.
static void assertEquals(java.lang.String message, boolean expected, boolean actual)


Datei Bearbeiten Ansicht Chronik Lesezeichen Extras Hilfe
Assert (JUnit API)
junit.sourceforge.net/javadoc/
Suchen
All Classes
Packages
org.hamcrest.core
org.junit
org.junit.matchers
org.junit.runner
org.junit.runner.manipulation
org.junit.runner.notification
org.junit.runners
All Classes
After
AfterClass
AllOf
AllTests
AnyOf
Assert
Assume
Before
BeforeClass
BlockJUnit4ClassRunner
ComparisonFailure
Describable
DescribedAs
Description
Failure
Filter
Filterable
Ignore
Is
IsAnything
IsEqual
IsInstanceOf
IsNot
IsNull
IsSame
JUnit4
JUnitCore
JUnitMatchers
NoTestsRemainException
Parameterized
Parameterized.Parameters
ParentRunner
Request
Result
RunListener
Overview Package Class Tree Deprecated Index Help
PREV CLASS NEXT CLASS
FRAMES NO FRAMES
SUMMARY: NESTED | FIELD | CONSTR | METHOD
DETAIL: FIELD | CONSTR | METHOD
org.junit
Class Assert
java.lang.Object
└ org.junit.Assert
public class Assert
extends java.lang.Object
A set of assertion methods useful for writing tests. Only failed assertions are recorded. These methods can be used directly: Assert.assertEquals(...), however, they read better if they are referenced through static import:
import static org.junit.Assert.*;
...
assertEquals(...);
See Also:
AssertionError
Constructor Summary
protected Assert()
Protect constructor since it is a static only class
Method Summary
static void assertArrayEquals(byte[] expecteds, byte[] actuals)
Asserts that two byte arrays are equal.
static void assertArrayEquals(char[] expecteds, char[] actuals)
Asserts that two char arrays are equal.
static void assertArrayEquals(int[] expecteds, int[] actuals)
Asserts that two int arrays are equal.
static void assertArrayEquals(long[] expecteds, long[] actuals)
Asserts that two long arrays are equal.
static void assertArrayEquals(java.lang.Object[] expecteds, java.lang.Object[] actuals)
Asserts that two object arrays are equal.
static void assertArrayEquals(short[] expecteds, short[] actuals)
Asserts that two short arrays are equal.
static void assertArrayEquals(java.lang.String message, byte[] expecteds, byte[] actuals)
Asserts that two byte arrays are equal.
static void assertArrayEquals(java.lang.String message, char[] expecteds, char[] actuals)

# Testfalltabelle für die Methode `steuer()`

| Test-fall | Erwar-teter Status | Klasse | `int einkommen` (Parameter `expected`) | Ausgabe (Parameter `actual`) |
|---|---|---|---|---|
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

# Testfalltabelle für die Methode `steuer()`

| Test-fall | Erwarteter Status | Klasse | `int einkommen` | Ausgabe |
|---|---|---|---|---|
| E1 | Exception | Einwohner | -1 | Einkommen darf nicht negativ sein |
| E2 | ok | Einwohner | 0 | 1 |
| E3 | ok | Einwohner | 20 | 2 |
| E4 | ok | Einwohner | 25 | 2 |

# Testfalltabelle für die Methode `steuer()`

| Test-fall | Erwarteter Status | Klasse | `int einkommen` | Ausgabe |
|---|---|---|---|---|
| A1 | Exception | Adel | -1 | Einkommen darf nicht negativ sein |
| A2 | ok | Adel | 0 | 20 |
| A3 | ok | Adel | 10 | 20 |
| A4 | ok | Adel | 20 | 20 |
| A5 | ok | Adel | 253 | 25 |

# Testfalltabelle für die Methode `steuer()`

| Test-fall | Erwarteter Status | Klasse | `int einkommen` | Ausgabe |
|---|---|---|---|---|
| A1 | Exception | Adel | -1 | Einkommen darf nicht negativ sein |
| A2 | ok | Adel | 0 | 20 |
| A3 | ok | Adel | 10 | 20 |
| A4 | ok | Adel | 20 | 20 |
| A5 | ok | Adel | 253 | 25 |

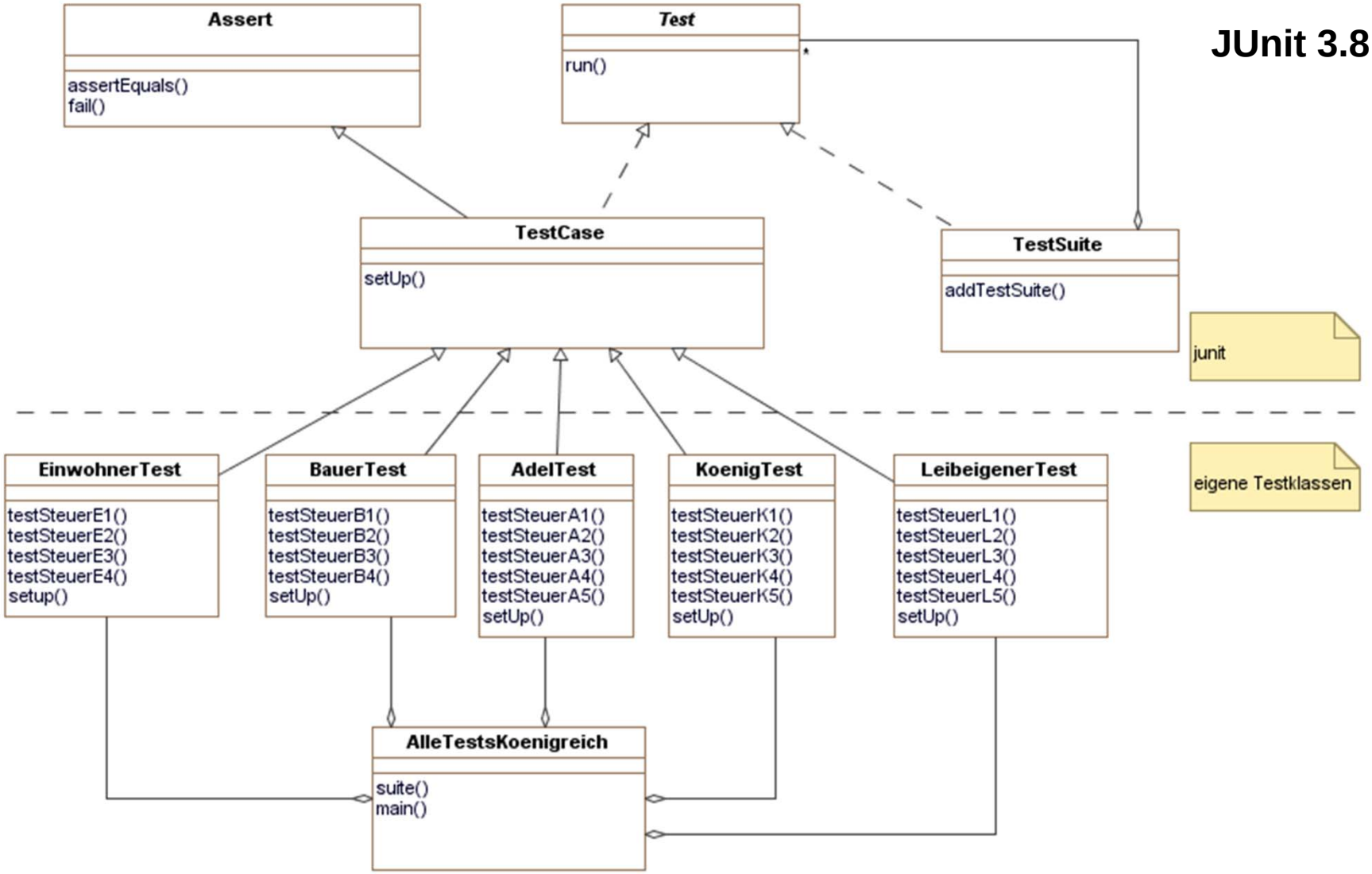
# Testfalltabelle für die Methode `steuer()`

| Test-fall | Erwarteter Status | Klasse | `int einkommen` | Ausgabe |
|---|---|---|---|---|
| B1 | Exception | Bauer | -1 | Einkommen darf nicht negativ sein |
| B2 | ok | Bauer | 0 | 1 |
| B3 | ok | Bauer | 20 | 2 |
| B4 | ok | Bauer | 25 | 2 |

# Testfalltabelle für die Methode `steuer()`

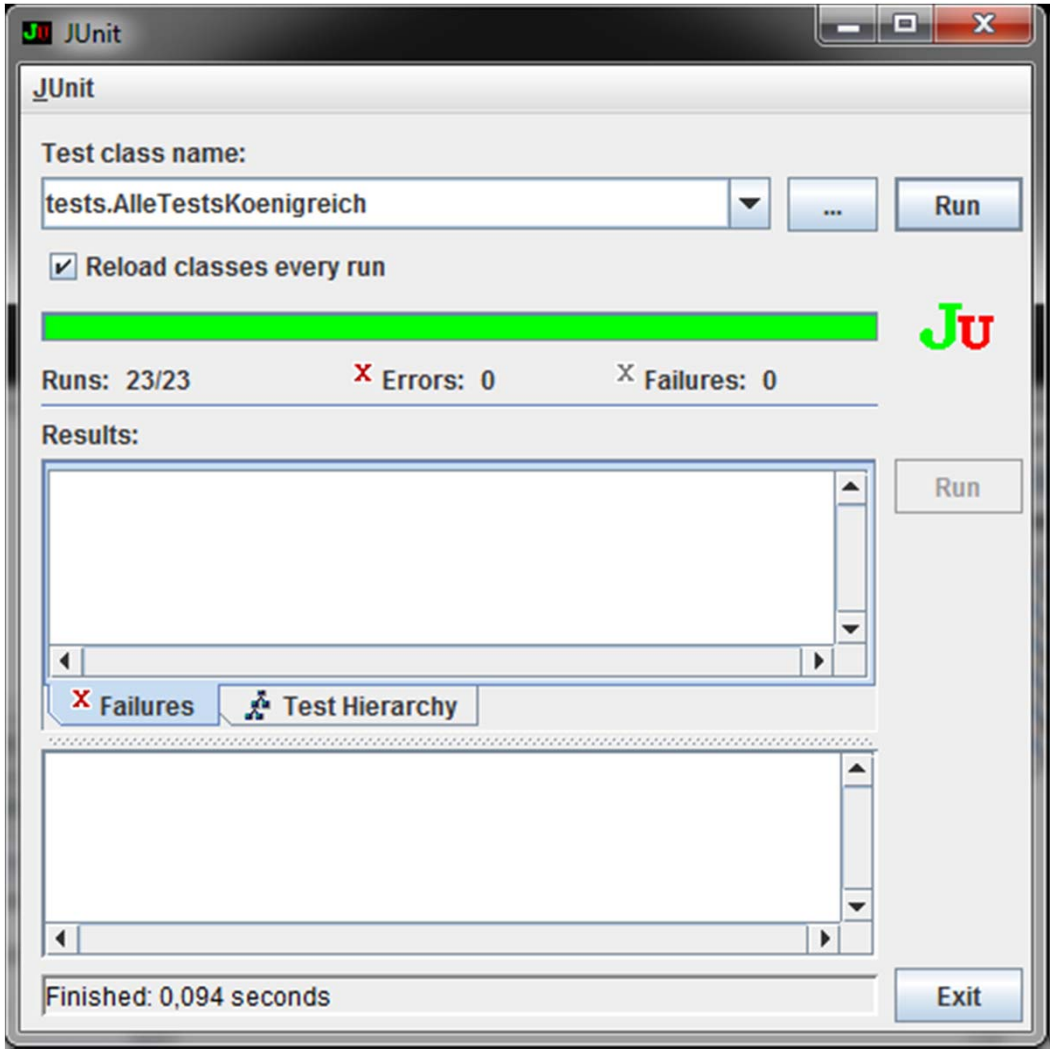
| Test-fall | Erwarteter Status | Klasse | `int einkommen` | Ausgabe |
|---|---|---|---|---|
| L1 | Exception | Leibeigener | -1 | Einkommen darf nicht negativ sein |
| L2 | ok | Leibeigener | 0 | 1 |
| L3 | ok | Leibeigener | 12 | 1 |
| L4 | ok | Leibeigener | 22 | 1 |
| L5 | ok | Leibeigener | 253 | 24 |

# JUnit 3.8.1


Assert
assertEquals()
fail()
Test
run()
*
TestCase
setUp()
TestSuite
addTestSuite()
junit
EinwohnerTest
testSteuerE1()
testSteuerE2()
testSteuerE3()
testSteuerE4()
setup()
BauerTest
testSteuerB1()
testSteuerB2()
testSteuerB3()
testSteuerB4()
setUp()
AdelTest
testSteuerA1()
testSteuerA2()
testSteuerA3()
testSteuerA4()
testSteuerA5()
setUp()
KoenigTest
testSteuerK1()
testSteuerK2()
testSteuerK3()
testSteuerK4()
testSteuerK5()
setUp()
LeibeigenerTest
testSteuerL1()
testSteuerL2()
testSteuerL3()
testSteuerL4()
testSteuerL5()
setUp()
eigene Testklassen
AlleTestsKoenigreich
suite()
main()

JUnit 3.8.1

# TESTVARIANTE 1




Erstellung einer Testsuite (AlleTestsKoenigreich) und Abarbeitung mit junit.swingui.TestRunner

**JUnit 3.8.1**

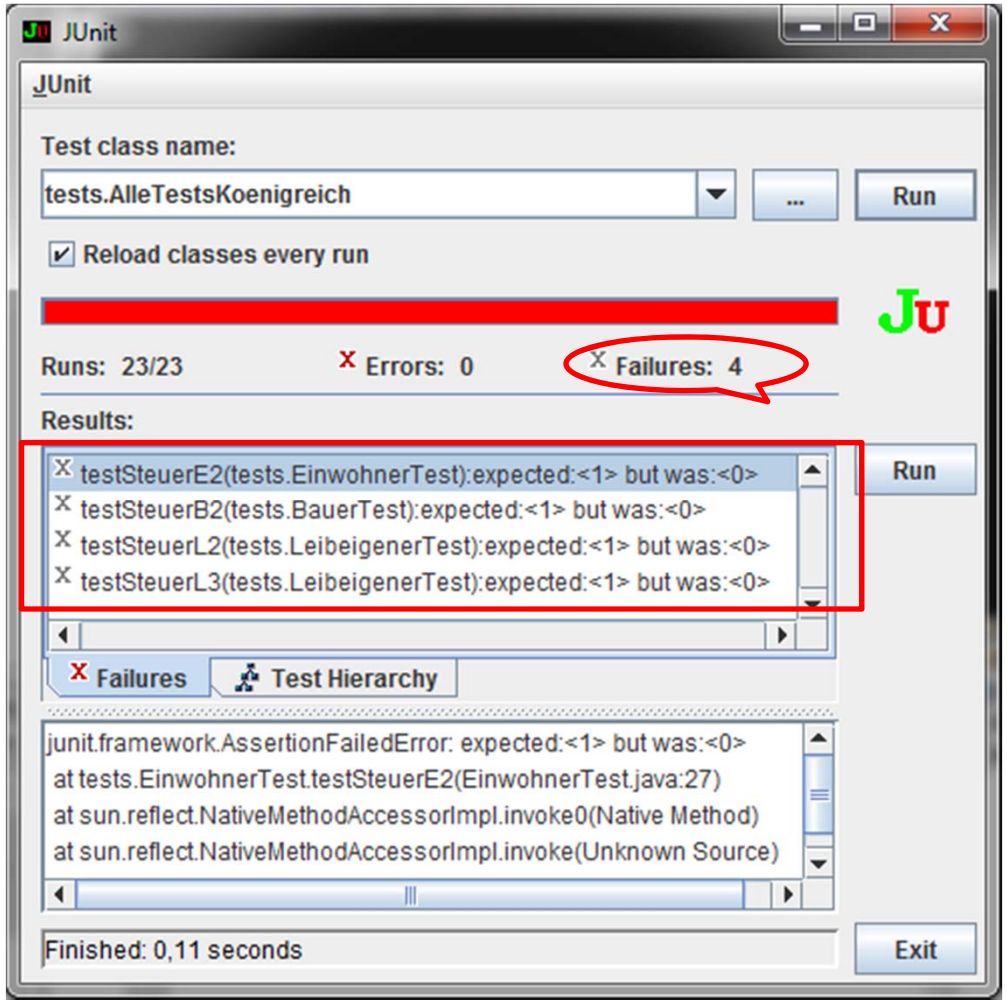
- **ok**-Fall (Gutfall): erfolgreicher Test (**Musterlösung: Test1**)

## Testfalltabelle mit Protokoll für die Methode `steuer()` (Musterlösung: Test1)

| Test-fall | Erwarteter Status | Klasse | `int einkommen` | Ausgabe | Zustand (Protokoll) |
|---|---|---|---|---|---|
| E1 | Exception | Einwohner | -1 | Einkommen darf nicht negativ sein | Grün |
| E2 | ok | Einwohner | 0 | 1 | Grün |
| E3 | ok | Einwohner | 20 | 2 | Grün |
| E4 | ok | Einwohner | 25 | 2 | Grün |

JUnit 3.8.1

# TESTVARIANTE 2




Hier wurde ein Failure in der Methode steuer() (Klasse Einwohner) provoziert .
Es wurde der Test auf die Mindeststeuer von 1 Taler vergessen.

**JUnit 3.8.1**

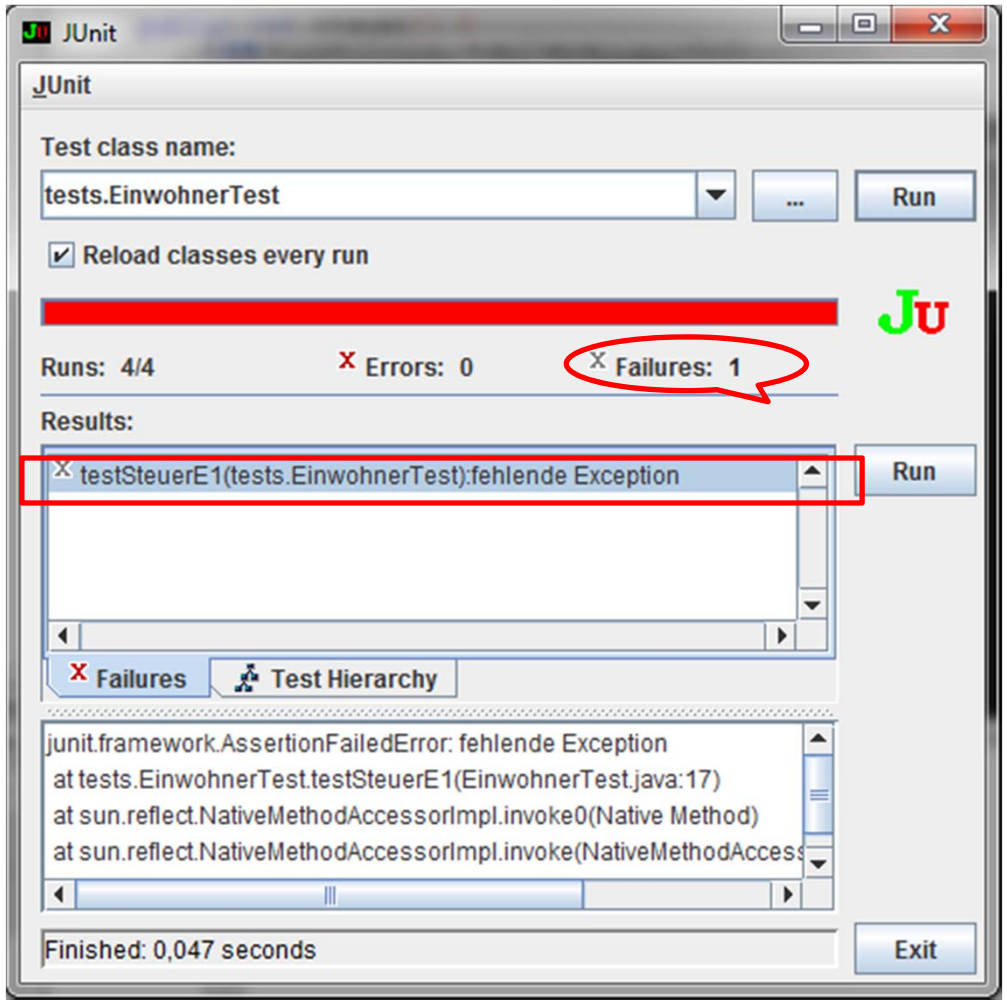
- **Failure**-Fall (Fehlerfall E2): Testfall, der scheitert (**Musterlösung: Test2**)

## Testfalltabelle mit Protokoll für die Methode `steuer()` (Musterlösung: Test2)

| Test-fall | Erwarteter Status | Klasse | `int einkommen` | Ausgabe | Zustand (Protokoll) |
|---|---|---|---|---|---|
| E1 | Exception | Einwohner | -1 | Einkommen darf nicht negativ sein | Grün |
| E2 | ok | Einwohner | 0 | 1 | Rot (Failure) |
| E3 | ok | Einwohner | 20 | 2 | Grün |
| E4 | ok | Einwohner | 25 | 2 | Grün |

JUnit 3.8.1

# TESTVARIANTE 3




Hier wurde ein Failure in der Methode setEinkommen() (Klasse Einwohner) provoziert :
Statt eine Exception zu werfen, wurde das Einkommen auf 0 gesetzt.

**JUnit 3.8.1**

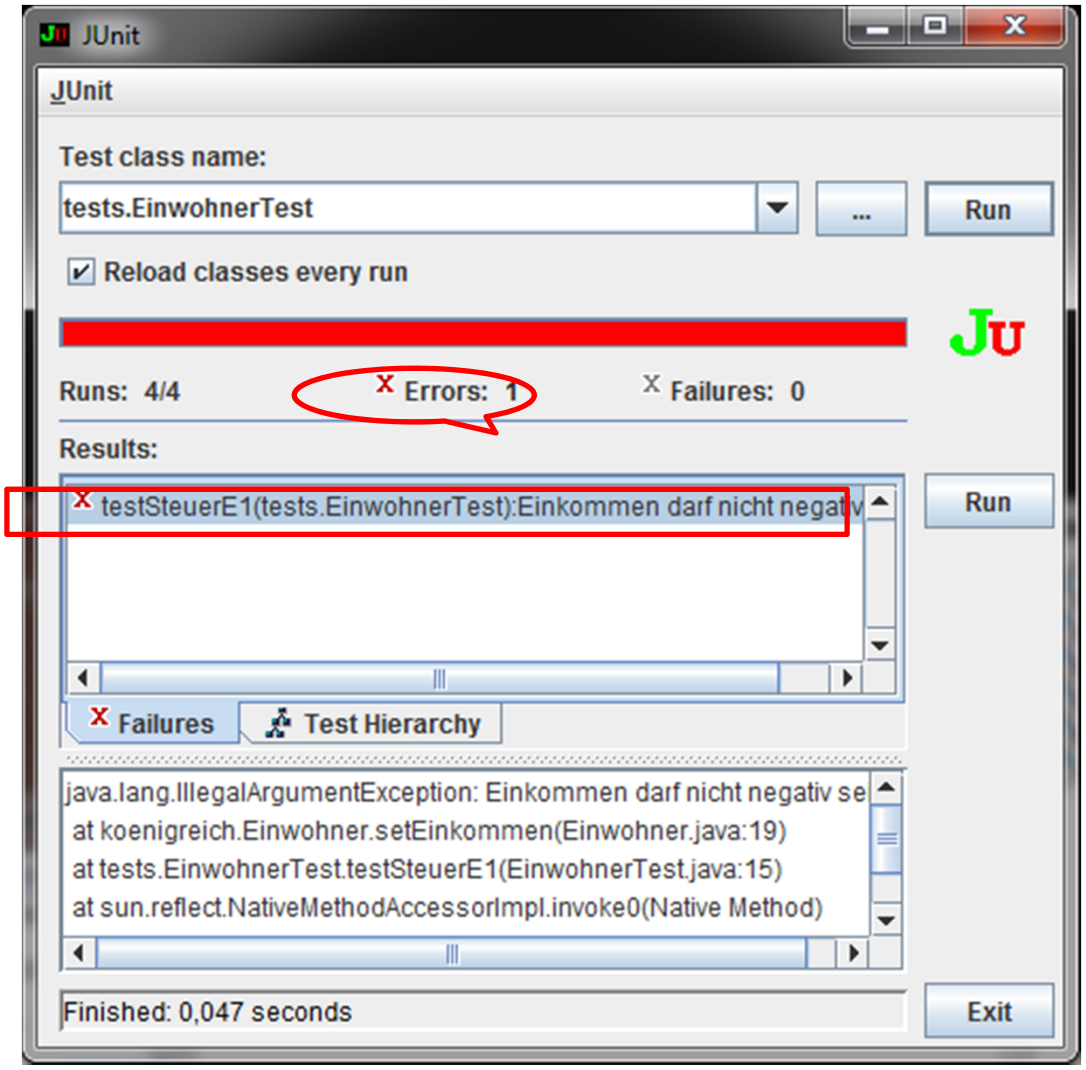
- **Failure**-Fall (Fehlerfall E1): Testfall, der scheitert (fail()) (**Musterlösung: Test3**)

## Testfalltabelle mit Protokoll für die Methode `steuer()` (Musterlösung: Test3)

| Test-fall | Erwarteter Status | Klasse | `int einkommen` | Ausgabe | Zustand (Protokoll) |
|---|---|---|---|---|---|
| E1 | Exception | Einwohner | -1 | Einkommen darf nicht negativ sein | Rot (Failure) |
| E2 | ok | Einwohner | 0 | 1 | Grün |
| E3 | ok | Einwohner | 20 | 2 | Grün |
| E4 | ok | Einwohner | 25 | 2 | Grün |

JUnit 3.8.1

# TESTVARIANTE 4




**Wir ändern die Anforderung:**

Bei einem negativen Einkommen soll laut Testfallspezifikation in der Methode setEinkommen() Einkommen auf 0 gesetzt werden, d.h. es soll keine Exception geworfen werden.

Es wird aber eine Exception geworfen.

**JUnit 3.8.1**

- **Error**-Fall (Exception): Testfall E1, der in einer Ausnahme endet (**Musterlösung: Test4**)

## Testfalltabelle mit Protokoll für die Methode `steuer()` (Musterlösung: Test4)

| Test-fall | Erwarteter Status | Klasse | `int einkommen` | Ausgabe | Zustand (Protokoll) |
|---|---|---|---|---|---|
| E1 | Exception | Einwohner | -1 | 1 | Rot (Error) |
| E2 | ok | Einwohner | 0 | 1 | Grün |
| E3 | ok | Einwohner | 20 | 2 | Grün |
| E4 | ok | Einwohner | 25 | 2 | Grün |