Freie Universität Berlin

Grundlagen des Software-Testens

Übung #5

Black-Box Tests

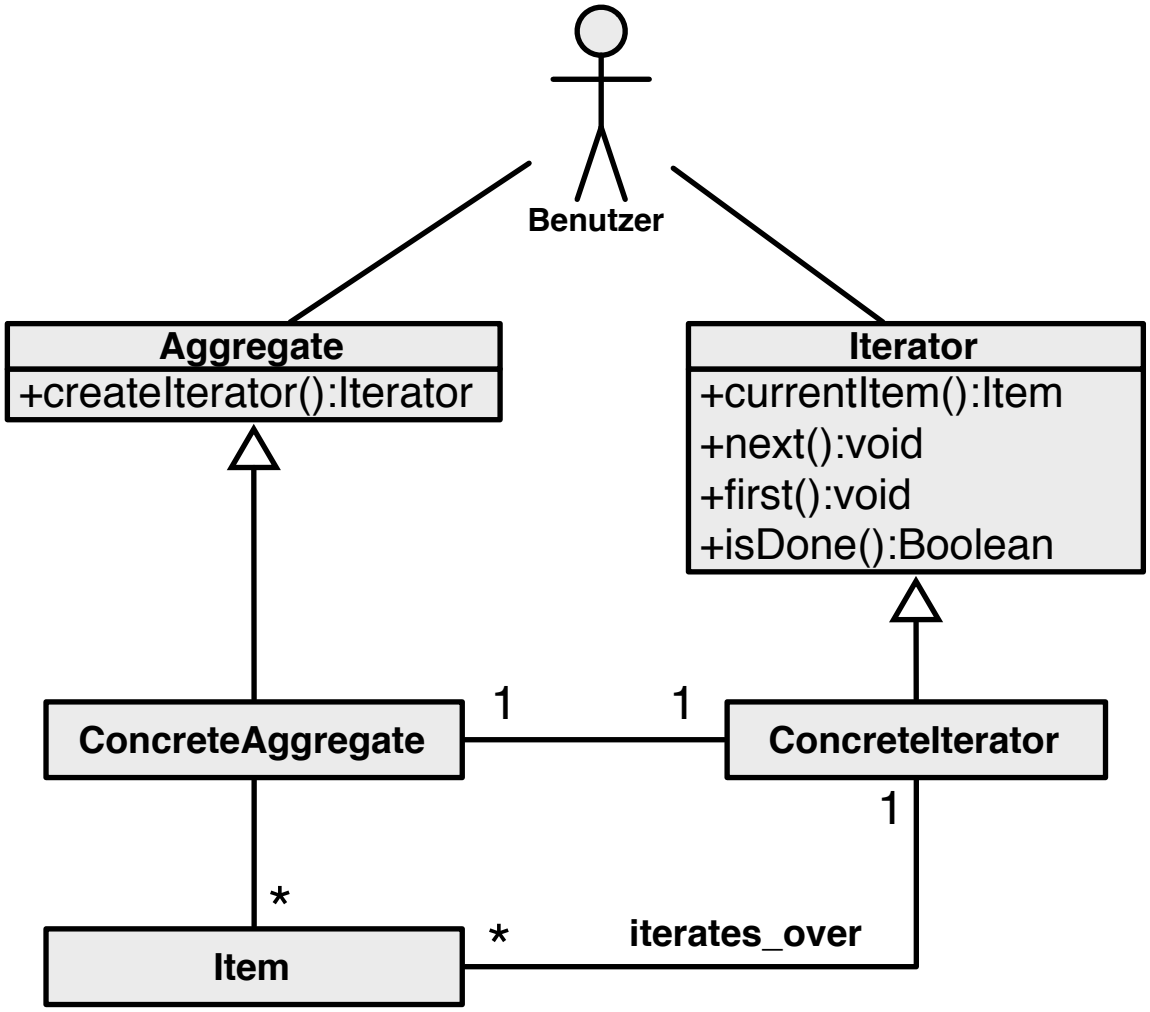
Ina Schieferdecker, Edzard Höfig
Fachgruppe Modellbasierte Entwicklung und Qualitätssicherung
Arbeitsgruppe Software Engineering
Institut für Informatik

Vorbereitung

- Ziel der Übung
  - Verständnis von **Black-Box** Testansätzen
    - Zustandsdiagramme
    - Übergangsbäume
    - Transitionsüberdeckung
  - Durchführung von **GUI Tests**

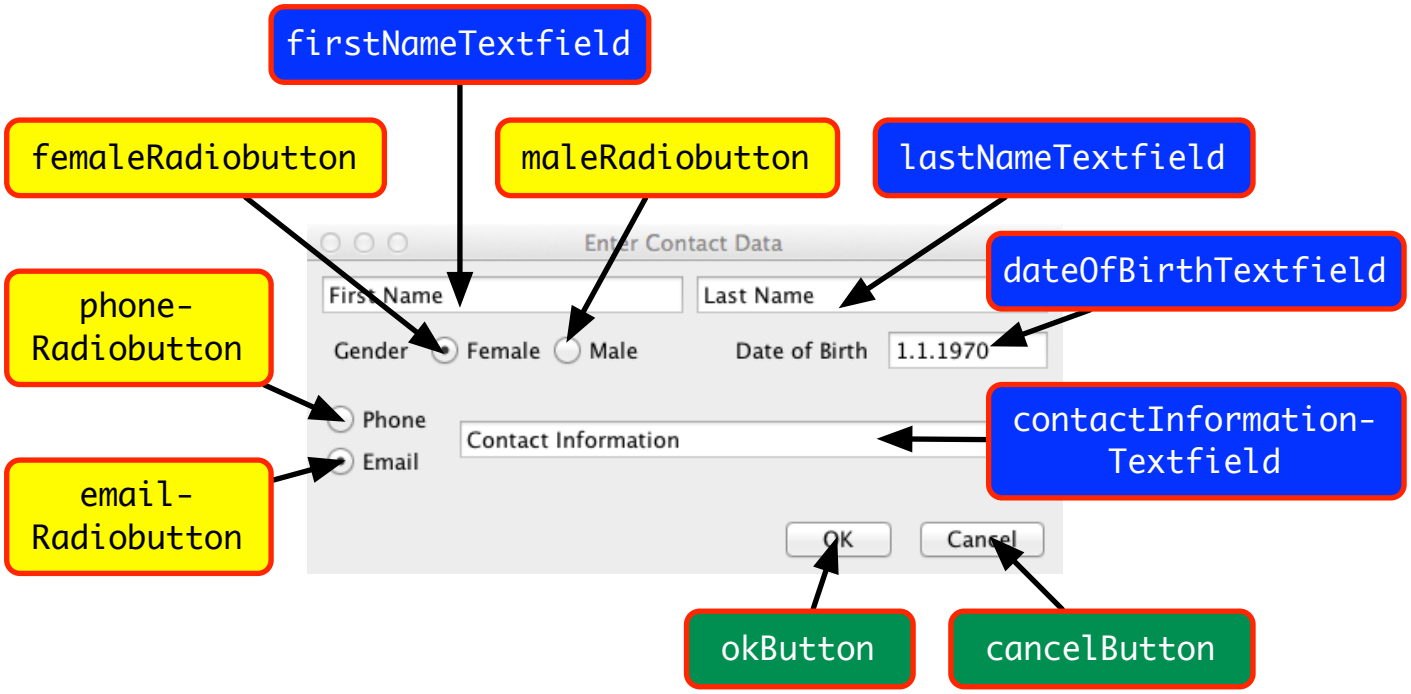
- Abgabe der Übungsaufgabe 5
  - Bis zum **Anfang der nächsten Übung**
    - Deadline ist am nächsten Freitag, den 02.09.2011 um 14h
  - Eine Abgabe pro Gruppe per **Blackboard**

Vorbereitung: Das Iterator Muster



Vorbereitung: GUI Tests mit Abbot

- Framework zur Automatisierung von GUI Tests
- Benötigt eindeutige Namen der Komponenten
  - Siehe **ComponentIDs.pdf**



Aufgabe 1 bis 3: Zustandsdiagramme

- Beschreibung der Aufgaben in **ZustandstestIterator.pdf**

- Aufgabe 1 *Zustandsdiagramm*
  - Entwerfen Sie ein Zustandsdiagramm für einen *Iterator*. Bei Ihren Überlegungen zu der Frage, wie viele und welche Zustände ein *Iterator* hat, sollten Sie stets überprüfen, ob Sie für jeden entworfenen Zustand feststellen können (durch Methoden-aufrufe), ob sich der *Iterator* in diesem Zustand befindet oder nicht. Ein nicht überprüfbarer Zustand ist in diesem Kontext unnütz. Denken Sie daran, sich für die einzelnen Zustandsübergänge zu überlegen, ob und welche Bedingungen für diesen Übergang erfüllt sein müssen. Gehen Sie zu diesem Zweck davon aus, dass `itemCount` die Anzahl der Elemente im Container bezeichnet. Benötigen Sie weitere Variablen? Gehen Sie ferner davon aus, dass ein *Iterator* jederzeit zerstört werden kann.

- Aufgabe 2 *Übergangsbaum*
  - Leiten Sie aus Ihrem Zustandsdiagramm aus Aufgabe 1 den zugehörigen Übergangsbaum ab.

- Aufgabe 3 *Tests zur Transitionsüberdeckung*
  - Leiten Sie logische Tests zur Transitionsüberdeckung bei Containern mit genau vier Elementen ab.

Aufgabe 4: GUI Test

- Verwenden Sie JUnit zur Überprüfung der korrekten Sortierreihenfolge beim Hinzufügen von Einträgen in das Adressbuch.
  - Testen Sie dabei **ausschließlich nach Black-Box Prinzipien** und greifen Sie niemals direkt auf Klassen zu die in den sub-packages `model`, `view` und `controller` des package `exercise5.addressbook` definiert sind.
  - Verwenden Sie das **Abbot Framework** zur Testdurchführung.
    - Eine Test-Fixture für Abbot ist bereits vorgegeben
- Hinweis:
  - Die aktuelle Version von Abbot (1.2.0) hat auf manchen Systemen (z.B. OS X 10.7) Schwierigkeiten die richtige "Keymap" zu erkennen. Als Folge davon werden einige Zeichen nicht richtig in die Textfelder eingetragen (z.B. Sonderzeichen, `y` und `z` vertauscht...).
  - Bitte überprüfen Sie bei Ihren Testfällen, ob Abbot die richtigen Testdaten einträgt und wählen Sie ggfs. andere.

Vorgehensweise für die praktische Aufgabe

1. Eclipse starten und das Übungsprojekt importieren
   - Das Übungsprojekt findet sich in Blackboard (**projekt5.zip**)
   - Es wird eine neue Bibliothek verwendet: *abbot*
2. Das Beispiel verstehen
   - Den Quellkode anschauen
   - Dazu die Klasse **exercise5.test.TestSorting** ausführen
3. TestFälle entwerfen und kodieren
   - In **exercise5/test/TestSorting.java**
   - Dazu die Dokumente **ComponentIDs.pdf** und **SpezifikationDerSortierung.pdf** lesen
4. Testfälle könnten Fehler finden!
5. Die fertigen Testklassen per Blackboard abgeben
   - Zusammen mit den theoretischen Aufgaben