LUXOFT TRAINING

Разработка через тестирование Integration Tests

Test Driven Development

Ivan Dyachenko <IDyachenko@luxoft.com>

Module 3

Для кого этот тренинг?

LUXOFT Training Center

1. **Beginner**
   Хорошая точка входа
2. **Intermediate**
   Поможет лучше всё структурировать в голове и объяснять коллегам
3. **Advanced**
   Можно использовать для обучения и проверки других

Содержание

LUXOFT Training Center

Интеграционное тестирование

Интеграционное тестирование — одна из фаз тестирования программного обеспечения, при которой отдельные программные модули объединяются и тестируются в группе.

Тестирование архитектуры системы

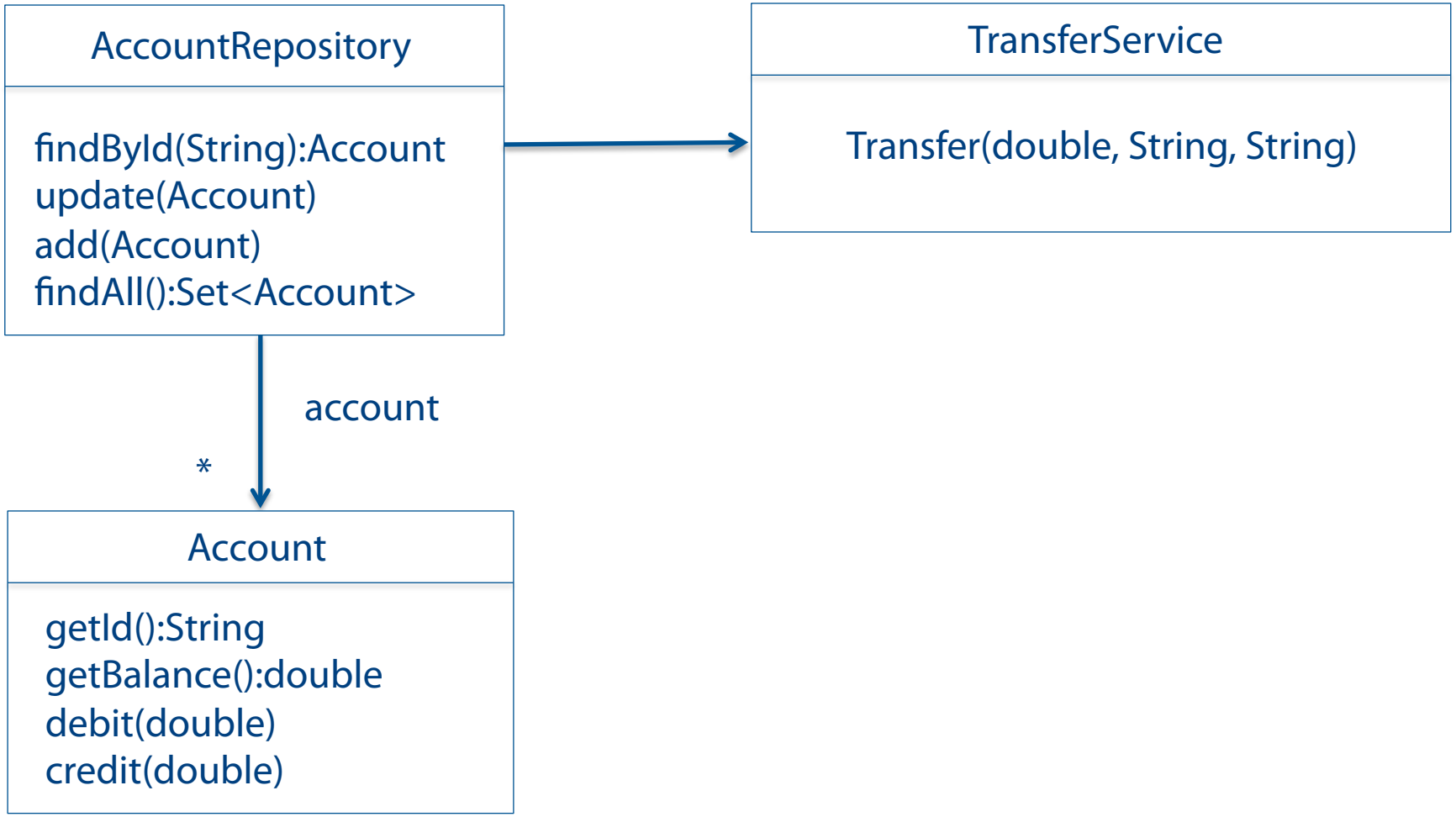
Интеграционное тестирование называют еще тестированием архитектуры системы.

Результаты выполнения интеграционных тестов – один из основных источников информации для процесса улучшения и уточнения архитектуры системы, межмодульных и межкомпонентных интерфейсов. Т.е. с интеграционные тесты проверяют корректность взаимодействия компонент системы.

Итеративный процесс

Интеграционное тестирование, как правило, представляет собой итеративный процесс, при котором проверяется функциональность все более и более увеличивающейся в размерах совокупности модулей.

Пример



JUnit

LUXOFT Training Center

В качестве инструмента мы все еще используем JUnit!

Чаще всего для написания интеграционных тестов используются те же библиотеки для тестирования, что и для модульных тестов.

Black Box Tesitng

Тестирование чёрного ящика или поведенческое тестирование — стратегия (метод) тестирования функционального поведения объекта (программы, системы) с точки зрения внешнего мира, при котором не используется знание о внутреннем устройстве тестируемого объекта.

Test-driving DB layer

LUXOFT Training Center

- Взаимодействие с источниками данных
- Интеграционные тесты на базу данных
- Транзакции в тестировании

Взаимодействие с источниками данных

Интеграционные тесты, которые изменяют данные в базе данных, должны откатывать состоянии базы данных к тому, которое было до запуска теста, даже если тест не прошёл.

Для этого часто применяются следующие техники:

- Метод TearDown, присутствующий в большинстве библиотек для тестирования.
- Try – catch - finally структуры обработки исключений, там где они доступны.
- Транзакции баз данных.
- Создание снимка (англ. snapshot) базы данных перед запуском тестов и откат к нему после окончания тестирования.
- Сброс базы данных в чистое состояние перед тестом, а не после них. Это может быть удобно, если интересно посмотреть состояние базы данных, оставшееся после не прошедшего теста.

Test asynchronous processes

LUXOFT Training Center

http://stackoverflow.com/questions/631598/how-to-use-junit-to-test-asynchronous-processes

Вопросы ?

Разработка через тестирование

IDyachenko@luxoft.com

git clone git://github.com/ivan-dyachenko/Trainings.git

https://github.com/ivan-dyachenko/Trainings