LUXOFT TRAINING

Разработка через тестирование Mock & Stub, EasyMock

Test Driven Development

Ivan Dyachenko <IDyachenko@luxoft.com>

Module 5

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

1

2

3

4

5

6

7

EasyMock

LUXOFT Training Center

- Одним из самых популярных инструментов, применяемых при interaction тестировании, является EasyMock
- Рассмотрим использование mock-объектов на примере простого клиент-серверного приложения SimpsonViewer, используемого для просмотра эпизодов Симпсонов

EasyMock

LUXOFT Training Center

У нас есть web-сервис, возвращающий требуемый эпизод (ISimpsonService)

EasyMock

Клиентская часть используется для получения эпизодов с сервера и показ их пользователю, являясь, по сути, клиентским прокси для web-сервиса

EasyMock

TBD

EasyMock

Напишем первый тест, который инициализирует наш сервис с null-параметром

EasyMock

LUXOFT Training Center

В текущей реализации тест провалится, поэтому изменим наш сервис чтобы тест был успешным

EasyMock

LUXOFT Training Center

- Однако конструктор не должен постоянно «кидать» исключение, а только в том случае, когда в качестве параметра передается null
- Напишем тест, который не должен падать, если мы передадим в конструктор реализацию ISimpsonService
- Поскольку у нас нет реализации сервиса (кроме используемой в приложении), будем использовать mock-объект

EasyMock

- TBD

EasyMock

- В текущей реализации тест провалится, поэтому изменим наш сервис чтобы тест был успешным

EasyMock

LUXOFT Training Center

- Перейдём к тестированию метода getEpisode()
- Здесь нам понадобится заглушки как для ISimpsonService, так и для IEpisode

EasyMock

LUXOFT Training Center

- Методом expext(), мы задаем, какие методы объекта и с какими параметрами должны быть вызваны
- Методом andReturn() можно задать возвращаемый результат

EasyMock

LUXOFT Training Center

- Метод replay() завершает настройку («запись») заглушки и переводит ее в режим использования («воспроизведения»)
- В режиме записи, mock-объект еще не является заглушкой, а лишь «записывает», что он должен делать

EasyMock

LUXOFT Training Center

- После вызова replay(), он начинает работать как заглушка
- Метод verify() как правило, завершает сценарий использования объекта и проверяет, действительно ли были сделаны требуемые вызовы с нужными параметрами

EasyMock

LUXOFT Training Center

- Поскольку у нас еще нет реализации метода getEpisode(), тест провалится
- Дополним реализацию нужным методом

EasyMock

LUXOFT Training Center

T.o. схема использования mock-фреймворка выглядит следующим образом:

- Создаем mock-объект
- Задаем ожидания вызовов и возвращаемые значения
- Переводим mock в режим «воспроизведения»
- Вызываем методы тестируемого объекта
- Проверяем, что наши ожидания соответствуют реальному поведению объекта

EasyMock

LUXOFT Training Center

- Если метод должен выполниться несколько раз, то задается это следующим образом:

TBD

- Для указания количества вызовов так же могут использоваться setter'ы: atLeastOnce(), anyTimes(), times(from, to)

EasyMock

LUXOFT Training Center

- Стоит помнить, что после вызова EasyMock.verify() для mock-объекта, он считается «отработанным»
- Для того, чтобы снова использовать этот объект нужно вызвать EasyMock.reset() и заново установить поведение mock-объекта

EasyMock

LUXOFT Training Center

- Тестируем исключения

EasyMock

LUXOFT Training Center

- В зависимости от целей тестирования, EasyMock предлагает несколько разновидностей mock-объектов
- Nice mock
  - В отличие от дефолтного mock'а, не вызывает AssertionError на незапланированные вызовы, а возвращает значение по умолчанию (0, null или false)
  - Создается вызовом EasyMock.createNiceMock()

Вопросы ?

Разработка через тестирование

IDyachenko@luxoft.com

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

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