テストパターンの有効性について

株式会社オーエスケイ
小井土 亨

自己紹介

- 株式会社 オーエスケイ
  - http://www.kk-osk.co.jp/
  - 株式会社 大塚商会のソフトウェア研究 開発拠点として1984年に100%子会社として設立
- 小井土 亨
  - 1984年 株式会社大塚商会入社
  - 同年子会社オーエスケイに出向し、現在に至る
    - パッケージソフトウェアの開発及び技術調査を担当
  - CBOP .NET分科会主査

アジェンダ

- 求められているソフトウェア開発
  - テストの重要性
  - テストファーストにおけるテスト
- 良いテスト
  - 良いテストと効果
- テストパターン
  - インフォメーションセンター
  - ビューステート
  - オーダーシート

求められているソフトウェア開発

- ビジネスの変化
  - 顧客のニーズに合わせて、ビジネスを柔軟に変化させることが成功の鍵
  - ビジネスの変化に合わせて、ソフトウェアも変更を行うことが求められている
  - 多くの変化は予測不可能

変化に強く 継続的で迅速なソフトウェア開発

継続的で迅速な開発に必要な項目

- 明確な要求定義
  - 早い時期での顧客による仕様確認
    - 継続的な短期リリース
- 変更コストを一定に保つ
  - プログラムを常に変更しやすい状態に保つ
    - シンプルな設計
    - リファクタリング 設計の改善）
- 高い品質
  - 機能の正しさ、実行速度、使い勝手など

当たり前ですが「テストが重要」

継続的なリリース
継続的なテストが必要

リファクタリング
自動テストによる確認が必須

高い品質
テストによる確認が必要

テストが開発を牽引

テストの問題点と解決策

- 問題点１
  - 計画が遅れて、テスト期間が確保できない
  - 解決策　→　テストファースト、Test Driven Development
- 問題点２
  - 手作業による人海戦術
  - 解決策　→　テストフレームワークの導入
- 問題点３
  - 品質の良いテストが書けない
  - 解決策　→　テストパターン

テストファーストにおけるテスト作成

- 最初は、必ずテストから
  - 仕様書的な役割を持つテストから作成を開始する
- テストと実クラスは、同じ担当者が作成する
  - テストとメソッドの実装を繰り返すことで、クラスに対する理解が深くなり、テストの品質とカバレッジ率が高くなる
- 作業は、少しずつ進める
  - メソッドのテストを作成しては、メソッドを実装する
- コンパイルしたら、必ず1度はテストする
  - コンパイルしたら、テストで動作を確認する
- テストの独立性を維持する
  - テストの独立性を維持すると実クラス間の独立性も維持されて、変更に強い構造となる

良いテストの条件

- テスト目的が明確である
  - 何をテストするか明確である
  - その目的も明確で、更にひとつであること
- テストの判定が正しい
  - テストの成功、失敗が正しく判断されている
- テストを独立して実行することができる
  - テストが他のテストに依存することなく独立している
- 繰り返し実行することができる
  - 何度でも繰り返して、テストを実行することができる
- テストを実行しても、状態が変化しない
  - テストを実行し、成功した場合でも失敗した場合でも、テストを実行する前と後で何も変わらない

良いテストの効果

- メソッドが一つの機能を実現している
  - メソッドが明確な一つの機能だけを提供する
- メソッドの結果を提供する
  - 外部に対して、メソッドの処理結果を判断できるような何らかの方法を提供する
- クラスやメソッドの独立度が高いこと
  - クラスやメソッドが、他のクラスやメソッドの依存が低い
- 特定の環境への依存度が低い
  - 特定のファイルやデータベース構造などに対する依存度が低い

重要なポイント

- シンプル
  - テスト、クラス、メソッドなど、全てがシンプルであることが最善である
  - 常にシンプルになっているか確認し、改善（リファクタリング）して、シンプルを追求する
- 低い依存度
  - テスト、クラス、メソッドなど、全てが他のテストやクラスやメソッドに対して、依存している部分を低くすることが重要である
  - 依存度を低くして、高い独立性を追求する

テストパターンとは

- テストパターンの目的
  - テストファーストにおいて効率的なテストを行うための知識を共有する
- 対象とするテスト
  - 自動化することができるテスト
  - 単体テスト（単体クラスレベルのテスト）
  - 総合テスト（シナリオレベルのテスト）
  - 受入れテスト（システムの外部機能レベルのテスト）
- テストパターンの範囲
  - テストのパターン
  - テスト及びテスト対象のクラスから構成されるパターン

インフォメーションセンター

Q 環境に依存するプログラムをテストしたい

A 事前にテスト用に環境を構築し、実行時に環境を切り替えてテストを可能とする
環境に依存する設定部分を一元管理する

- 利用場面
  - データベースの状態に依存するテストの実行
    - データベースの状態に依存するテストを行う場合に、状態を再現したデータベースに環境を変更してテストを実行する
  - 設定ファイルやファイルに依存するテストの実行
    - ファイルの内容に依存するテストを行う場合に、様々な内容のファイルを用意し、対象を変更してテストを実行する

ビューステート

Q ユーザーインターフェイスをテストしたい

A ユーザーインターフェイスの状態やロジックを別クラスに分離し、自動テストを行う

- 利用場面
  - ユーザーインターフェイスのテスト
    - ユーザーインターフェイスの状態やロジックを自動テストする
  - シナリオテスト
    - 複数のユーザーインターフェイスに対応したビューステートクラスを作成し、様々なシナリオテストを行う

オーダーシート

Q 結合度の高い複数のクラスをテストしたい

A クラス間の依存関係を無くして、テストを実行する

- 利用場面
  - 処理のバリエーション実行テスト
    - オーダーシートクラスに様々な設定を行い、カバレッジの高いテストの実行する
  - シナリオテスト
    - 複数のビューステートクラスにオーダーシートクラスを渡す方法で、シナリオ型のユーザーインターフェイステストを行う