60分で学ぶ実践E2Eテスト

2022/3/10 JaSST`22 Tokyo

末村拓也(Autify)・伊藤由貴(ベリサーブ)

セッション概要

内容

Webアプリケーションを題材に、テスト設計からテストコード実装までの自動テスト作成の流れを一気通貫で、実践的に解説していきます。

想定参加者

- ソフトウェアテストの経験がある方
- プログラミングの経験があり、簡単な環境構築くらいは出来るよ！という方

セッションの構成

大きく2部構成です

1. 設計編：どんなテストを自動化するのか、を考える
2. 実践編：設計編で考えたテストを、ツールを使って実際に自動化する

自己紹介：伊藤由貴

- テスト自動化エヴァンジェリスト @ベリサーブ
  - テスト自動化の普及活動を行っている
- コミュニティ活動
  - JaSST Tohoku実行委員
  - JSTQBテスト自動化エンジニアシラバス翻訳WG

自己紹介：末村拓也

- Technical Support Engineer @ Autify, Inc.
  - 色々なWebアプリとブラウザの相性問題を解決していく仕事
- JaSST Online 実行委員

設計編

はじめに：皆さんに質問

「E2Eテストを自動化しよう」と思った場合、何から始めますか？

- 思いつくままに自動化する
- 今ある手動テストを順番に自動化する
- なんとなく大事そうなテストケースから自動化する

アンチパターンです

本パートの概要

- やみくもにE2Eテストを自動化するのは危険
- 何をテストするのか、どこから自動化するのかを考える必要がある
    - この考え方を説明

※正確には「何をテストするのか」はJSTQBで言うところの「テスト分析」が相当。

やみくもにE2Eテストを自動化するデメリット

- 自動テストが増え、メンテナンスが辛くなる
- 自動テストが増え、実行時間（期間）が延びる
- 実行しても不具合を見つけられないテストが増える

などなど

何をテストするか、を考える際のポイント

- 単体テストや結合テストなど、自組織の各テストがどんな役割を担っているか
  - その中でE2Eテストでは何を担保したいか
- 理想とする開発サイクル
  - リリース頻度、CI/CDパイプライン、テストにかけられる時間、など

Point：「テストの自動化」に閉じずに、開発・テストをどうしたいかを考えよう

テストレベルごとの役割

- テストピラミッドにおいては、Unit, API, UIの3層
  - 日本で一般には「単体テスト」「結合テスト」「システムテスト」のような区分で呼ばれる
  - 組織によって呼び名は様々
- 自分たちが「E2Eテスト」と呼ぶテストでは何を担保したいのか、を決める

参考：「E2Eテスト」とは

- システムテスト？ユーザー受け入れテスト？UIテスト？
  - 組織によって定義が異なる
  - そして、業界統一の「正しい定義」もない
- 本セッションでは、システムをユーザーと同じように操作して行うテストで、かつユーザーストーリー単位で行うものをE2Eテストと呼ぶことにする

※組織でのE2Eテストの定義が本ページと異なっていても、実践編でテストを自動化する際の技術要素は共通しているため、何らかお役には立つと思います

ここからは具体的なサイトに対してE2Eテストを考えてみましょう

ホテル予約サイトのE2Eテストを考えよう

- テスト対象：HOTEL PLANISPHERE - テスト自動化練習サイト
- 条件設定
  - サイトは公開済み
  - 機能追加やBugfix後のリリース前にリグレッションテストとしてE2Eテストを行っているが、現在は手動
  - E2Eテストを自動化して効率化したい

各テストレベルで行っていることを確認

- 単体テスト
  - 金額計算ロジックのみ整備
  - hotel-example-site/billing_test.html
- 結合テスト
  - なし

E2Eテストで行うことを考える

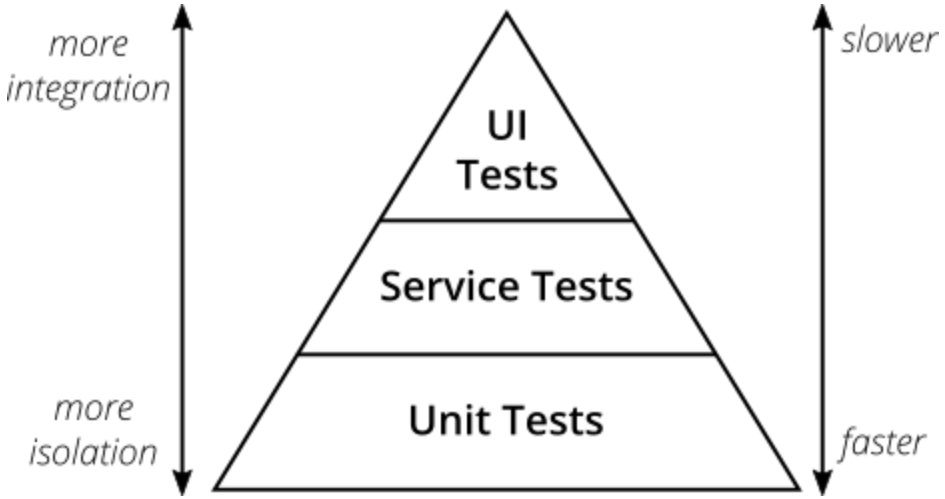
- 単体テストや結合テストで行っている・行えるテストはそちらでやるべき
  - テストピラミッドの考え方
- ではE2Eでしかできない範囲とは？
  - 画面描画が関係するテストなど
- 基本的なユーザストーリーは最低限網羅しておきたい

参考：一般的にE2Eテストとして自動化されるもの

- ユーザへの影響度が高い機能やユーザシナリオ
  - 起動/終了、ログイン/ログアウト、課金・購入、など
- ビジネス上の影響度が高い機能やユーザシナリオ
  - 会員登録、メルマガなど

などを、複雑な異常系や多数のデータパターンではなく正常系を中心に実行することが多い

参考：テストピラミッド



- UI Tests（今回のE2Eテストはここに相当）は、実行に時間がかかる
- Unit Testは実行にかかる時間が少ない
- 下の層でテストできるものは下の層でやったほうが効率がよい
    - なんでもかんでもE2Eテストで画面からテストするのは非効率

E2Eテストの自動化対象を選ぶ考え方

- ユーザーが使う各機能について、基本的な正常系のユーザーシナリオをテストする
- 「ここをカバーしておけばユーザーの大半に影響がない」と言える範囲をカバーする
  - 人数ベース：IE11を使っているのは3%だから対象外にしよう
  - 金額ベース：ガラケーを使っているのは全体の5%だけど、売上の30%を占めているから対象にしよう

今回対象にする範囲

- 重要機能
  - ログイン・ログアウト、予約、会員登録
  - 特に予約機能はサイトの主たる機能のため、会員種別ごとに予約可否を確認
- E2Eでしかできないテスト
  - 会員種別ごとのプラン表示確認
  - 例）一般会員向けの画面にプレミアム会員専用のプランが表示されていないこと

選定したユーザーシナリオ6つ

1. 非会員で予約
2. 会員登録→予約→ログアウト
3. プレミアム会員でログイン→予約→ログアウト
4. 一般会員でログイン→予約→ログアウト
5. 一般会員の画面にプレミアム会員限定プランが表示されないこと
6. 非会員の画面に一般・プレミアム会員限定プランが表示されないこと

このうち、1番目の「非会員で予約」のシナリオを、具体的な手順として書き起こします

非会員で予約するシナリオの手順(1/2)

1. https://hotel.testplanisphere.dev/ja/ を開く
2. メニューから「宿泊予約」を選択
3. 宿泊プラン一覧から「お得な特典付きプラン」の「このプランで予約」を選択
4. 宿泊日を翌月1日に設定
5. 宿泊数を7泊に設定
6. 人数を2に設定
7. 朝食バイキング、昼からチェックインプラン、お得な観光プランを選択
8. 氏名に「テスト太郎」を入力
9.

非会員で予約するシナリオの手順(2/2)

9. 確認のご連絡をメールに設定
10. メールアドレスにhoge@hoge.conを設定
11. ご要望・ご連絡事項に「テスト」と入力
12. 予約内容を確認するボタンを選択
13. 宿泊予約確認画面で、以下を確認
    i. 合計金額が123,000円であること
    ii. 期間、人数、追加プラン、お名前、確認のご連絡、ご要望・ご連絡が入力通りになっていること
14. この内容で予約するボタンを選択し、以下を確認
    i. 予約が完了しましたダイアログが表示されること

手順化のポイント

- 実行のたびに結果が変わらないようにする
  - 休日と平日とで料金が異なるため、単純に翌月1日から宿泊にしただけでは金額が可変になってしまう。7泊に設定することで、いつ実行しても同じ金額になる
- 「任意の〜」は避ける
  - 自動テストを実装する際に困る

ここからは、選定＆手順化したテストケースについて、実際にコードを書いて自動化していきましょう