実践で学ぶ、効率的な自動テストスクリプトのメンテナンス

2013.12.01

伊藤望(STAR)

講師紹介

- TODO!!!

システムテスト実行の自動化

- GUI(画面)自動テストツール
- 画面操作を自動化し、テスト作業を効率化!
- Selenium, QTP, UWSC, など様々なツールがある
- ブラウザ・モバイルのテストツール
- オープンソース

このハンズオンで学ぶこと

1. Seleniumの基本的な使い方
2. Seleniumテストを効率よくメンテナンスする方法

いろいろなSelenium ①

- Selenium IDE
  - ブラウザ操作の記録と再生

いろいろなSelenium ②

- Selenium WebDriver
  - プログラミング言語のコードから実行

効率よくテストをメンテナンスするなら

- Selenium IDE
  - 手軽にテストを作れる
- Selenium WebDriver
  - 長期にわたってメンテナンスし続けるならこちら
- 今日はSelenium WebDriverについて学びます

タイムテーブル

1. Selenium WebDriverの使い方	
1-1. 入門課題	40分
1-2. 実践課題	30分
休憩	
2. Selenium WebDriverテストを効率よくメンテナンスする	
2-1. 概要説明	20分
2-2. 実践課題:ページオブジェクトデザインパターン	80分
休憩	
2-3. 実践課題:システムのバージョンアップ	40分

1. Selenium WebDriverの使い方

1-1. 入門課題 (40分)

入門課題

- Selenium WebDriverの基礎を学びます
- 5分程度のミニ課題×7
- 必要なもの
  - Eclipse
  - Google Chrome
  - ハンズオン用インストールキット

入門課題その1
「動かしてみよう、Selenium」

1. Eclipseを起動します
2. test/introwork/IntroWork1.javaを開いてください

入門課題その1
「IntroWork1.java」を実行し、 成功することを確認してください

- 手順
  1. test/introwork/IntroWork1.javaを右クリックし、「実行」>「JUnitテスト」を選びます
  2. テストが実行され、結果が緑になれば成功です

入門課題その1　解説
JUnit

- テストの実行には、テストフレームワーク「JUnit」を使っています
- @Before
  - 初期処理
- @Test
  - メインとなるテスト処理
- @After
  - 終了処理

入門課題その1　解説
@Before

```
@Before
public void setUp() {
    // chromedriverのインストール場所を指定
    System.setProperty(
        "webdriver.chrome.driver",
        chromeDriverPath());

    // WebDriverのインスタンスを生成しブラウザを起動
    driver = new ChromeDriver();
}
```

入門課題その1　解説
@Test

```
@Test
public void test() {
    ……

    // 指定したURLのウェブページに移動
    driver.get(url);

    // 文字列入力・クリックなどの処理
    ……
}
```

入門課題その1　解説
@After

```
@After
public void tearDown() {
    // ブラウザを閉じ、WebDriverを終了する
    driver.quit();
}
```

入門課題その2
「クリックしてみよう」

1. test/introwork/IntroWork2.javaを右クリック し、「実行」>「JUnitテスト」を選びます
2. 「OK」ボタンが置かれたページが表示されます

入門課題その2
「OK」ボタンをクリックする処理を、IntroWork2.javaに実装してください

入門課題その2
「クリックしてみよう」

- 「OK」ボタンのidを調べます
  1. introWork2.htmlをGoogle Chromeから直接開きます
  2. 「OK」ボタンを右クリックし「要素の検証」を選びます
- Sleep処理を消して、クリック操作を記述します

```
WebElement okButton
     = driver.findElement(By.id("要素のid"));
okButton.click();
```

- 書けたら実行してみます

入門課題その2
「クリックしてみよう」

- 動きが速すぎて、クリックできたか分からない時は
  1. driver.quitにブレークポイントを置きます
  2. IntroWork2.javaを右クリックし、「デバッグ」>「Junitテスト」からテストを実行します
  3. ブレークポイントでテストが一時停止するので、クリックできたか確認できます

入門課題その3
「文字列を入力してみよう」

1. IntroWork3.javaをJUnitテストとして実行します
2. テキスト入力欄が置かれたページが表示されます

入門課題その3 (5分)
テキスト入力欄に「自動テストカンファレンス」という文字列を入力する処理を、IntroWork3.javaに実装してください

- ヒント

```
WebElement input
    = driver.findElement(By.id("要素のid"));
input.sendKeys("文字列");
```

入門課題その4
「ラジオボタンを選択してみよう」

- IntroWork4.javaをJUnitテストとして実行すると、課題ページが表示されます

入門課題その4 (5分)
ラジオボタンの「あり」の選択肢を選ぶ処理を、IntroWork4.javaに実装してください

- ヒント
  - ラジオボタンの選択は「click」で行います

入門課題その5
「チェックボックスを選択してみよう」

- IntroWork5.javaをJUnitテストとして実行すると、課題ページが表示されます

入門課題その5 (5分)
チェックボックスのチェックをオンにする処理を、IntroWork5.javaに実装してください

入門課題その5
「チェックボックスを選択してみよう」

- ヒント
  - チェックボックスのチェックの切り替えは「click」で行います
  - 既にチェック状態なら、チェックを切り替えないようにします

```
if (!element.isSelected()) {
    element.click();
}
```

入門課題その6
「プルダウンを選択してみよう」

- IntroWork6.javaをJUnitテストとして実行すると、課題ページが表示されます

入門課題その6 (5分)
プルダウンの選択値を5にする処理を、IntroWork6.javaに実装してください

入門課題その6
「プルダウンを選択してみよう」

- ヒント

```
Select headCount = new Select(
    driver.findElement(By.id("要素のid")));
headCount.selectByValue("5");
```

入門課題その7
「表示された値のチェックをしてみよう」

- IntroWork7.javaをJUnitテストとして実行すると、課題ページが表示されます

入門課題その7 (5分)
表示された金額の値が「8000」であることをチェックする処理を、IntroWork7.javaに実装してください

入門課題7
「表示された値のチェックをしてみよう」

- ヒント
  - getTextにより表示されているテキストを取得
  - JUnitのAssertThatメソッドを使って値をチェック

```
import static org.junit.Assert.*;
import static org.hamcrest.core.Is.*;

……

WebElement total
    = driver.findElement(By.id("要素のid"));
assertThat(total.getText(), is("値"));
```

入門課題で学んだこと

- クリック
- 文字列入力
- ラジオボタン
- チェックボックス
- プルダウン
- 値チェック

1. Selenium WebDriverの使い方

1-2. 実践課題 (30分)

実践課題その1

- 入門課題で学んだ内容を実践します
- test/practicework/PracticeWork1.javaをJUnitテストとして実行すると、「STARホテル宿泊予約画面」が表示されます

実践課題その1 (30分)
docs/TestCase.pdfの「実践課題その1」テストケースを、PracticeWork1.javaに実装してください。 • 予約処理の自動化 • 確認画面の値チェックの自動化

実践課題その1 ヒント

- 既に値が入っているテキスト入力欄に文字列を入力する場合、先に値をクリアする必要があります。

```
element.clear();
element.sendKeys("文字列");
```

2. Selenium WebDriverテストを効率よくメンテナンスする

2-1. 概要説明 (20分)

色々なSelenium

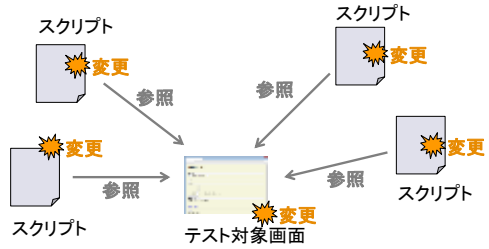
- Selenium IDE
  - ブラウザ操作の記録と再生
- Selenium WebDriver
  - プログラミング言語のコードから実行

Selenium IDE

- キャプチャ&リプレイツール
- メリット
  - プログラムが書けなくても、短時間でテストスクリプトが作成できる
- デメリット
  - 作ったスクリプトのメンテナンス作業が大変

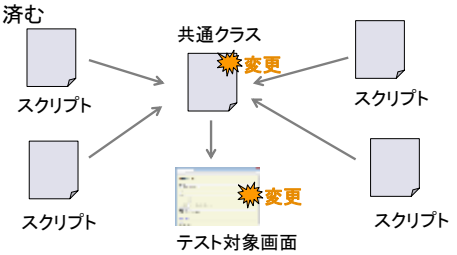
Selenium IDE スクリプトのメンテナンス

- テスト対象画面に変更があると大変



Selenium WebDriver

- 画面が変わるとスクリプトの修正が必要な点は同じ
- プログラムの共通化をうまく行えば、修正は1か所で済む



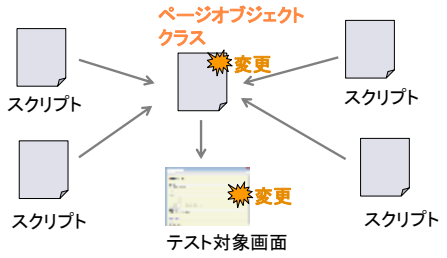
色々なSelenium まとめ

	テストが簡単に作成できる	共通化により、メンテナンスコストを抑えられる
Selenium WebDriver	×	○
Selenium IDE	○	×

- Selenium IDE
  - 手軽にテストを作れる
- Selenium WebDriver
  - 長期にわたってメンテナンスし続けるならこちら

Selenium WebDriver プログラムの共通化

- ページオブジェクトデザインパターン



「STARホテル宿泊予約画面」のページオブジェクト

```
public class ReserveInputPage {
    ……
    public void setReserveYear(String value) {
        ……
    }

    public void setReserveMonth(String value) {
        ……
    }
    ……
}
```

「STARホテル宿泊予約画面」のページオブジェクト

```
public class ReserveInputPage {
    ……
    public void setReserveYear(String value) {
        WebElement element = driver.findElement(
            By.id("reserve_year"));
        element.clear();
        element.sendKeys(value);
    }

    public void setReserveMonth(String value) {
        ……
    }
    ……
}
```

ページオブジェクトを使ったスクリプト

```
ReserveInputPage inputPage
    = new ReserveInputPage(driver);
inputPage.setReserveYear("2013");
```

- idなどのHTML情報が、スクリプト中に現れない
- click、sendKeysなどのWebDriverの処理もスクリプト中に現れない

2. Selenium WebDriverテストを効率よくメンテナンスする

2-2. 実践課題:ページオブジェクトデザインパターン (80分)

ページオブジェクトデザインパターンを実践

- 実践課題その2
  - 「実践課題その1」テストケースをページオブジェクトで書き換える
- 実践課題その3・4(早く終わった方はチャレンジ!)
  - ページオブジェクトを使ってテストケースを実装する練習

実践課題その2

実践課題その2 (40分)

次の3つの実装を完成させてください。

- 1ページ目「予約入力画面」のページオブジェクト
  test/practicework/pages/ReserveInputPage.java
- 2ページ目「予約確認画面」のページオブジェクト
  test/practicework/pages/ReserveConfirmPage.java
- 「実践課題その1」テストケースをページオブジェクトで実装し直した、test/practicework/PracticeWork2.java

実践課題その2 ヒント

- 朝食バイキングの値のsetメソッド

```
public void setBreakfast(boolean on) { …… }
```

- ページ遷移
  - ページ遷移を起こすメソッドの返り値を別のページオブジェクトにする

```
ReserveConfirmPage confirmPage
    = inputPage.goToNext();
```

実践課題その3・4

- 「実践課題その2」が早く終わった方はチャレンジ!

実践課題その3 (10分)

「実践課題その3」テストケースをページオブジェクトで実装した、test/practicework/PracticeWork3.java を完成させてください。

実践課題その4 (30分)

「実践課題その4」テストケースをページオブジェクトで実装した、test/practicework/PracticeWork4.java を完成させてください。

実践課題その3 ヒント

- 要素が存在するかどうかを調べる方法

```
driver.findElements(…).size > 0
```

実践課題その4 ヒント

- テキスト入力欄の値の取得

```
driver.findElement(…).getAttribute("value")
```

- ラジオボタンの選択状態の取得

```
driver.findElement(…).getAttribute("checked")
```

- チェックボックスのチェック状態の取得

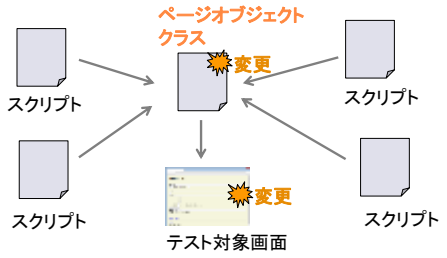
```
driver.findElement(…).isSelected()
```

2. Selenium WebDriverテストを効率よくメンテナンスする

2-3. 実践課題:システムのバージョンアップ (40分)

テスト対象画面が変更された時の影響範囲

- ページオブジェクトデザインパターン



実践課題その5

- 実際にテスト対象画面が変更されると、どんな修正が必要になるか、体感してみましょう。
- 実践課題その2で作成した、test/practicework/PracticeWork2.javaを開きます
- URLを`"reserveApp/index.html"`から`"reserveApp_Renewal/index.html"`に書き換えます
- PracticeWork2.javaを実行し、失敗することを確認します。

実践課題その5

実践課題その5 (30分)

PracticeWork2.javaのURLを
`"reserveApp_Renewal/index.html"`
に書き換えたテストが成功するよう、ページオブジェクトの内容を書き換えてください。

- ヒント
  - 書き換え前のページオブジェクトは、バックアップを取っておくのがお勧めです。

実践課題その5 ヒント

- setReserveDateメソッドの実装

```
element.sendKeys(
        year + "/" + month + "/" + day);
```

+

```
element.sendKeys(Keys.RETURN);
```

実践課題その6

実践課題その6 (10分)

PracticeWork3.javaとPracticeWork4.javaのURLを
`"reserveApp_Renewal/index.html"`
に書き換えたテストが成功するよう、ページオブジェクトの内容を書き換えてください。

本日のまとめ

- Selenium WebDriverの基礎を学びました
- ページオブジェクトデザインパターンを学びました
  - 変更されやすい画面情報を1ヶ所に集約して、効率よくメンテナンス