Selenium WebDriverでWebアプリのテストが変わる（前編）：

iPhone／Android含むブラウザ自動テストの最終兵器Selenium WebDriverとは

https://www.atmarkit.co.jp/ait/articles/1210/05/news104.html

Chrome、Firefox、Internet Explorer、Opera、Android、iOSといったブラウザに対応し、Java、C#、Python、Rubyなどが使えるWebテスト自動化ツールの3つの特徴と環境、実装方法を簡単に紹介する【2017年の情報に合うように更新】。

2017年07月19日 05時00分 更新

[後藤正規, ビーブレイクシステムズ]

Webのテストに超絶便利な「Selenium WebDriver」とは

Webアプリケーションのテスト自動化をサポートするツール「Selenium WebDriver」は2011年にリリースされました。

Selenium WebDriverは広範なWebブラウザのサポートを行っていた「Selenium1（Selenium RC）」と高速軽量で汎用的なWebブラウザエミュレータの機能を持つ「WebDriver」を統合したものです。

本稿では、Selenium WebDriverを簡単に試してみたい方や自動テストの実施を検討している方のために、前後編に分けて紹介します。Selenium WebDriverの特徴を整理するとともに、Selenium WebDriverを利用したWebアプリケーションに対する簡単な自動テストの実装、実施手法について解説します。

今回の主な内容

本稿で使用する用語の説明

- **Selenium WebDriver**

Selenium WebDriverはSelenium 1.xを引き継いだものです。本稿では、Webブラウザをコントロールするライブラリ群、または、その仕組みを指します。

- **WebDriver**

Selenium WebDriverには、「FirefoxDriver」「ChromeDriver」など、各Webブラウザのコントロール用のAPIが、それぞれ実装されており、その各実装を「WebDriver」とします。

3つの大きな特徴

Selenium WebDriverには、大きく下記3点の特徴があります。

1. **複数言語に対応**
2. **複数Webブラウザ対応**
3. **設計・実装の明確さ**

テストプログラムの言語サポートが複数あり、それぞれの言語で複数Webブラウザをサポートしていいます。図1のテスト実行時イメージを参照してください。

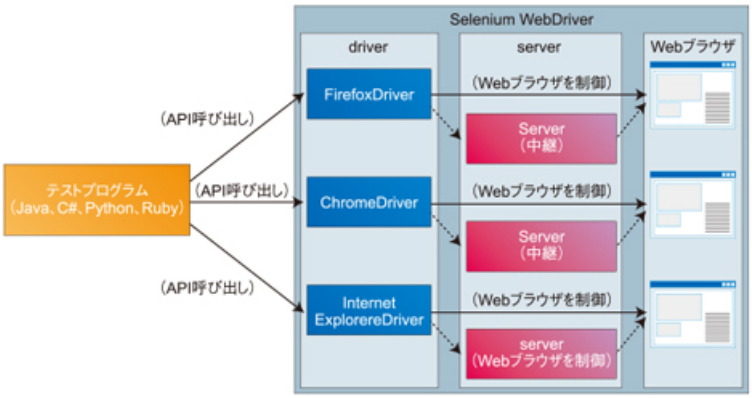

図1　Selenium WebDriverのテスト実行イメージ

【1】Java、C#、Python、Ruby、JavaScript（Node.js）などの複数言語に対応

　Selenium WebDriverを用いてテストを実行する際は、テストプログラムからWebDriver（図1ではdriver）のAPIを使用することで、Webブラウザをコントロールできます。

　Selenium WebDriverではテストプログラムの言語としてJava、C#、Python、Ruby、JavaScript（Node.js）の5言語を公式にサポートしています。

　なお他にも、Facebookなどのサードパーティーが PHPなど別の言語のサポートを行っています。

【2】iPhone／Android含む複数Webブラウザ対応

　また、テスト対象のWebブラウザに合わせて、使用するWebDriver（図1ではdriver）を変更してテストプログラムを実行します。現在サポートしているWebブラウザはChrome、Firefox、Internet Explorer、Microsoft Edge、Opera、Android標準Webブラウザ、iPhone標準Webブラウザ「Safari」です。

　図1では、サーバを中継して、Webブラウザをコントロールするような矢印も記述しています。これは下記の場合にが該当します。

- コントロール対象のWebブラウザがテストプログラムと同一PCではない場合
- 「Selenium-Grid」と呼ばれるテストケースの並列分散実行のツールを使用する場合
- テストプログラムをJava以外のプログラミング言語で実装し、かつHtmlUnit Driver（GUIを使用しないで、Webブラウザのテストを実行するWebDriver）を用いてテストを実行する場合

【3】設計・実装の明確さ

　第3の特徴である設計の分かりやすさ、実装しやすさに関しては図2に示してある各driverと親クラスとの継承関係のイメージを参照してください。

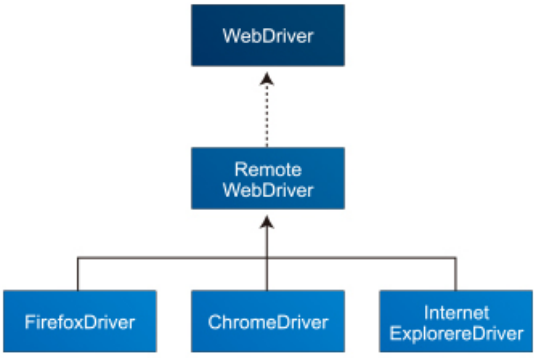

図2　WebDriverとWebブラウザごとの関係イメージ

　継承関係を見て分かるようにWebDriverを実装し、Webブラウザごとに特別にAPIが異なる場合を除いて、なるべく共

通のAPIを使用できるように設計されています。

もちろん、Webブラウザ特有の処理を実装しなければならない箇所もありますが、全体から見るとそれほど多くありません。これは複数のWebブラウザに対応する必要があるプロジェクトにとって非常に有意義であるといえるでしょう。

Selenium WebDriverを使うための準備

ここまでは、Selenium WebDriverの特徴について整理しました。ここからは、実装方法を紹介していきます。

まず、サンプルを用いた実装の紹介の前に準備と前提環境について説明します。

Selenium WebDriverのダウンロード

Selenium WebDriverはDownloadページの「Selenium Client & WebDriver Language Bindings」の項にあるテストプログラムを実装する対象言語のDownloadリンクからダウンロードします。

次に、テストするブラウザに合わせたWebDriverをダウンロードします。「Third Party Drivers, Bindings, and Plugins」の項から、対象ブラウザ用のWebDriverを取得してください。WebDriverの詳しい設定方法などについては同ページ内からリンクされているwikiなどを参照ください。

なお、Internet Explorer用のWebDriverについては「The Internet Explorer Driver Server」の項にあるので、こちらから取得してください。32bit用と64bit用のWebDriverがありますが、64bit版はかなり低速なため、基本的には32bit用のWebDriverを利用した方がよいでしょう。

前提環境の概要

以降、サンプルの解説においては下記の環境（表1）で行っている前提とします。

表1　前提環境

カテゴリ	ソフトウェア	バージョン
OS	Windows	7 Professional 64bit
Java	Java SE	8 Update 91
Web／アプリサーバ	Tomcat	8.5.14
IDE	Eclipse	4.6.2
Webブラウザ	Firefox	53.0
	Chrome	57.0.2987.133
	Internet Explorer（64bit版）	11.0.6900.18638
Selenium WebDriver	（Java用）	3.3.1

次ページでは、Selenium WebDriverのテストプログラムの記述方法について簡単に説明します。

Selenium WebDriver実装方法の概説

Selenium WebDriverを利用したプログラムは冒頭で説明したように、WebDriverのAPIを用いてWebブラウザを操作していきます。

例えば、下記のサンプルコードはFirefox用のWebDriverを利用して、テスト用のWebサイトを開き、閉じる処理を行っています。サンプルコードはJavaを使用しています。

```
package open;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;

public class EmployeeManagerOpen {

    public static void main(String[] args) {
        WebDriver driver = new FirefoxDriver();//…【1】
        driver.get("http://localhost:8080/EmployeeManager/index.jsp");//…【2】
        driver.quit();//…【3】
    }
}
```

サンプルコード1　EmployeeManagerOpen

まず、サンプルコード1の【1】を実行すると、図3のようにFirefoxが起動することが確認できます。

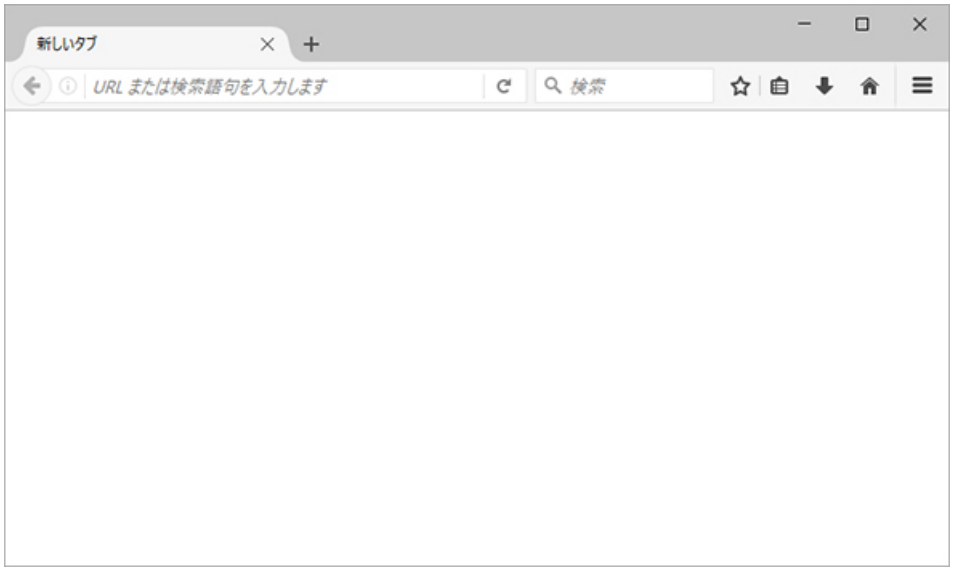

図3　Firefox用のWebDriverによって起動したFirefox

続いて【2】のコードが実行されると、図4のように引数のURLのデータを【1】で起動したWebブラウザで読み込みます。

従業員ID	名前	所属組織	性別	論理削除フラグ
000555	渡辺	営業部	男	OFF
012345	伊藤	管理部	男	OFF
111111	伊藤	営業部	女	OFF
222222	山本	開発部1部	女	OFF
333333	伊藤	開発部2部	男	OFF
444444	中村	営業部	女	OFF
555555	小林	開発部1部	男	OFF
666666	斎藤	管理部	男	OFF

図4　Firefox用のWebDriverによって読み込まれた従業員一覧画面

さらに【3】が実行されると、起動していたFirefoxが閉じることを確認できます。

このようにWebDriverは、Webブラウザの起動→（Webブラウザに対する個別処理）→Webブラウザの終了という一連の流れを実行できます。

JUnitも一緒に使う

サンプルコード1ではmainメソッドから記述していますが、JUnitを利用して下記（サンプルコード2）のような記述もできます。

```
package open;

import org.junit.After;
import org.junit.Before;
import org.junit.Test;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;

public class EmployeeManagerOpenTest {

    private  WebDriver driver;

    @Before
    public void beforeTest() {
        driver = new FirefoxDriver();
    }

    @Test
    public void open() {
        driver.get("http://localhost:8081/EmployeeManager/index.jsp");
    }

    @After
    public void afterTest() {
        driver.quit();
    }
}
```

サンプルコード2　EmployeeManagerOpenTest

Webブラウザ操作用のAPI一覧

　また、サンプルコード1、2では画面を開くAPI（get()メソッド）のみを使用していますが、org.openqa.selenium.WebDriverでは、その他にもWebブラウザ操作用のAPIが幾つか用意されています（表2参照）。

表2　JavaのWebDriver APIの概要

返り値	メソッド	処理概要
void	close()	現在handleしているWebブラウザウィンドウを閉じる
org.openqa.selenium.WebElement	findElement(By by)	現在のWebブラウザページ内において、引数の手法でWebElementを探し、最初に取得されたものを返却
java.util.List<WebElement>	findElements(By by)	現在のWebブラウザページ内において、引数の手法でWebElementを探し、取得されたもの全てをListに詰めて返却
void	get(String url)	引数で指定されたURLを読み込む
java.lang.String	getCurrentUrl()	現在のWebブラウザページで表示しているURLを表す文字列を取得
java.lang.String	getPageSource()	最後に読み込んだページのソースを取得
java.lang.String	getTitle()	現在のブラウザページで表示しているURLを表す文字列を取得
java.lang.String	getWindowHandle()	現在のWebブラウザウィンドウを表す文字列を表示（この文字列は複数ウィンドウを開いている場合などに、遷移する先のウィンドウ指定用に使用）
java.util.Set<java.util.String>	getWindowHandles()	現在開いている全てのWebブラウザウィンドウを表す文字列を取得
org.openqa.selenium.WebDriver.Options	manage()	オプションインタフェースを取得
org.openqa.selenium.WebDriver.Navigation	navigate()	Webブラウザの履歴情報へアクセス
void	quit()	Webブラウザウィンドウを全て閉じる
org.openqa.selenium.WebDriver.TargetLocator	switchTo()	WebDriverの管理対象を別フレーム、または別ウィンドウへ移動用のインタフェースを取得

　表では、JavaにおけるWebDriverの大まかな概要のみを記述しています。詳細な情報はSelenium WebDriverの各言語APIドキュメントを参照してください。

- Java
- C#
- Ruby
- Python
- JavaScript（Node.js）

後編は、サンプルアプリとテストケース

ここまで、サンプルを動かす上での前提環境と実装方法の概要を説明しました。後編は、サンプルアプリケーションとテストケースについて説明します。

■更新履歴

【2012/10/5】初版公開（後藤正規、株式会社ビーブレイクシステムズ）。

【2017/7/19】2017年の情報に合うように更新（今泉俊幸、株式会社ビーブレイクシステムズ）。

筆者紹介

今泉 俊幸（いまいずみ としゆき）

2011年から、株式会社ビーブレイクシステムズに在籍。

快適に仕事をするためには教育と自動テストこそが大事、という思いを抱き活動中

関連記事

Selenium VBAを使って自動でブラウザーを操作してスクショをExcelに張り付けてみた
システム開発におけるソフトウェアテスト（結合テスト〜システムテスト）において重要視されるエビデンス（作業記録）。前後編の2回にわたって、エビデンスとしてスクリーンショットをキャプチャし、テスト仕様書や納品書に張り付けていく作業を自動化するためのVBA／マクロのテクニックを紹介する。前編はSelenium VBAのインストール方法と使い方、スクリーンショットを自動でExcelに張り付ける方法について。

SeleniumのUIテスト自動化をiOS／AndroidにもたらすAppiumの基礎知識とインストール方法、基本的な使い方
本連載では、AndroidおよびiOSアプリ開発における、システムテストを自動化するツールを紹介していきます。今回は、オープンソースのモバイルテスト自動化ツール「Appium」の特徴やインストール方法、基本的な使い方を説明します。

JavaScriptテストの基礎知識と使えるフレームワーク6選
しっかりとJavaScriptの“テスト”を行うために、最近のJavaScript事情やテストを取り巻く環境、今注目のテストフレームワークを6つ紹介する