## 最速でJavaエンジニア になって稼ぐ

2018/12/22

人生逃げ切りオンラインサロン内





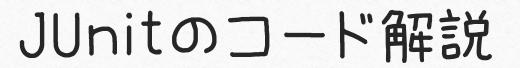
#### 目次的なやつ

- JUnitのコード解説
- 画面自動テストとは
- Selenide実践と解説













## GitHubのテストコードを見ながら 一つ一つ解説。





日付計算アプリケーションのサンプル https://github.com/koujienami/DateCalculation

Spring Bootでテストを書くときのやりかたまとめ https://qiita.com/mitsuya/items/be50dc329b4f3abe5ac5

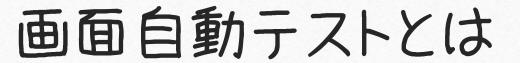
Spring Bootとユニットテスト環境の設計について https://qiita.com/rubytomato@github/items/0baf1df5b3e b7094cc41

mybatis-spring-boot-starter 1.3の変更点 https://qiita.com/kazuki43zoo/items/3f94d2049d331f81e 6f5



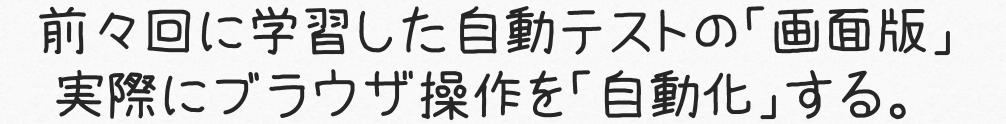




















- 主に結合テストや機能単体テスト、というようなある程度の規模があるもので利用される。
- 画面からの操作になる為、少なくとも画面上で動作していないとテスト実行すら出来ない。
- テスト駆動開発で画面自動テストを使ってやるケースもあるが、難易度が高く、またスピードもそこまで出ない。
- メリットとしては、ブラウザから実行するので、実際の操作と同等の事を自動化する事が出来る。ユーザーの行う操作をそのままテストケースにする事も可能なので、実に即していて、かつ、プログラムの課題で難しい要素である「見える化」が容易。
- デメリットも多くある。それなりにテスト作成に手間がかかること、実行時間も多く取られること、ブラウザ実行の都合上、環境に依存しやすいことなど。





利害とは全く関係ないけど、画面自動化は動いている様が面白い。それは結構楽しい。









- 画面自動化テストの為に利用するツールは「Selenium」を拡張したJava向けのテスティングフレームワーク「Selenide」。
- SeleniumはWebDriverというブラウザが提供している機能と連携して、ブラウザ操作を行うものだけど、そのまま使うと非常に使いにくい。やりたい操作を直感的に扱えない上に、コード量も増えるので、かなりしんどい。
- その為、Seleniumをラップしたものを使う事が殆ど。今回はSelenide。言語によっても違いがあり、それぞれでテストフレームワークが存在。
- またSeleniumやSelenideがある程度吸収していたり、ブラウザ側が良い感じに協調路線を走っていたりするものの、WebDriver自体が各ブラウザから提供されているという事もあって「ブラウザ毎の挙動の違い」が起きる。
- こればっかりはどうしようもないので、マルチブラウザ対応っていう要件が入ってきた場合は、テストケースやパターンが増える事は覚悟しよう。
- IEは滅んで欲しい。









Selenide実践と解説





日付計算アプリケーションに画面自動化をするためのテストコードを追加する。









日付計算アプリケーションのサンプル https://github.com/koujienami/DateCalculation

Selenide入門 https://qiita.com/EichiSanden/items/ea857b46cbf5435b0 c3b

Selenideノウハウ https://qiita.com/OSKD/items/27881470ea4f93790f32

Java製SeleniumラッパーのSelenideを使ってみた https://qiita.com/nyasba/items/6ab42fc73a912426ee5d







#### 次回までの宿題





# 引き続きアプリケーションのテストコードを追加。画面自動化も含め。











