入門！自動結合テスト

kyon_mm

2013/11/1 jasst' 13 Kyushu

Web Automate Integration Testing

自己紹介

- きょん(@kyon_mm)
- ソフトウェアテストアーキテクト
- うさみみ系エンジニア
- Groovy, C#, F#, Ruby, Scala
- SCMBootCamp, TDDBootCamp, 基礎勉強会, Nagoya.Testing, Cafe.Testing, STAR, TDDeXchange

アジェンダ

- 私の背景(Webアプリ中心とかチームが優秀とか)
- 品質を保つ自動テスト(どの立場で関わるのか大切)
- テスト設計について(共有しやすい形とか)
  - 詳細はイブニングセッション！
- 取り組み方（Groovy系，C#系，Scala系)
  - 詳細はイブニングセッション！

私の背景

- 開発対象
  - .NETで動くサーバーサイドのメッセージ型のWebアプリケーション。RESTfulAPI的な。
  - たまにGUIもあり。
  - HTTPでXMLとかJSONを交換する感じ。

私の背景

- 開発規模
  - 1ヶ月から3ヶ月くらいが多い。大きいと1年。
- 開発人数
  - 社内チームはマネージャー含めて2 - 4人が多い。
  - 他ベンダーと協調して作る事が多い。

私の背景

- 開発者
  - 優秀だと言い張れるくらいには優秀
  - お互いを尊敬し合いながら意見交換できる
  - 綺麗なコードが正義であり、自動化された単体テストがないプロダクトはあり得ないという文化

アジェンダ

- 私の背景(Webアプリ中心とかチームが優秀とか)
- 品質を保つ自動テスト(どの立場で関わるのか大切)
- テスト設計について(共有しやすい形とか)
- 取り組み方 (Groovy系, C#系, Scala系)

品質を保つ自動テスト

- 【まえがき】テストのコスト意識はとても重要です。テストのコスト意識はとても重要です。

品質を保つ自動テスト

- テストの自動化は何度も実行しなければもとが取れないとかいう話があります。

品質を保つ自動テスト

- 自動化は３回やらない
  と元がとれない？
- 目を覚ませ。建前はい
  らないのだよ。

品質を保つ自動テスト

- テストの自動化は何度も実行しなければもとが取れないとかいう話がありますが、そういうのは建前です。嘘です。いい訳です。

- 「結合テスト自動化で得られる最大のメリットはテスト実装者が得る幅広いプログラミングスキルとアーキテクチャ知識である」

- 「手動では不可能なテストの実装、コストの大幅低減」を実現するのは多くはシステムテストレベルである事が（比較的）多い。

品質を保つ自動テスト

- 「結合テスト自動化で得られる最大のメリットはテスト実装者が得る幅広いプログラミングスキルとアーキテクチャ知識である」

- 結合テストレベルの自動化をしなくていいと言っているのは、上のメリットを「(優先順位を考慮して)必要ない」と言っているのと同義だと捉えていると考え直してから決定する。

品質を保つ自動テスト

- 自動テストを誰かが勝手にやってくれるものとして保証する
- 自動テストを自分の手足のように使う(理解する)ものとして保証する
- 自動テストなしで保証する
- どの立場でテストを行うかはあなた次第

アジェンダ

- 私の背景(Webアプリ中心とかチームが優秀とか)
- 品質を保つ自動テスト(どの立場で関わるのか大切)
- テスト設計について(共有しやすい形とか)
- 取り組み方 (Groovy系, C#系, Scala系)

テスト設計について

- テストプロセスで重要なもの
  - 見積もり
  - 優先順位の変更
  - 実装のスケール
  - 保守
  - 実施のスケール

テスト設計について

- EmacsのOrg-Mode表記（個人的にオススメ
- UserStory + 6W2H
- マインドマップ（個人的にオススメ
- マトリクス
- オブジェクト図

アジェンダ

- 私の背景(Webアプリ中心とかチームが優秀とか)
- 品質を保つ自動テスト(どの立場で関わるのか大切)
- テスト設計について(共有しやすい形とか)
- 取り組み方 (Groovy系, C#系, Scala系)

取り組み方

- Groovy
  - Spock : 表形式とコメントを豊富につけられるテスティングフレームワーク
  - AsyncHttpClient : Httpのクライアントとして高性能な非同期通信ライブラリ
  - Geb : WebDriverをベースにしたWebGUI操作のテストライブラリ

取り組み方

- C#
  - TestAPI : 因子水準の組み合わせ、豊富な値ランダム生成などを持っているライブラリ
  - WebHttpClient : 標準のWebClient
  - NUnit : C# のテスティングフレームワーク
  - EntityFramework : ORマッパー

取り組み方

- Scala
  - Gatling : HTTPでの性能テストフレームワーク。普通のWebClientなテストフレームワークとしても使える。
  - ScalikeJDBC : DB操作のためのORマッパー