

RSpecによる
Railsアプリケーション
BDD事例

Y u g u i

今日のお題

✻ Ruby on Rails

✻ RSpec

✻ BDD

✻ Selenium統合

✻ 「こんな開発をしたチームがあるよ」

今日のお題

すごく、普通です

ギーク → @ujihisa

ま ず

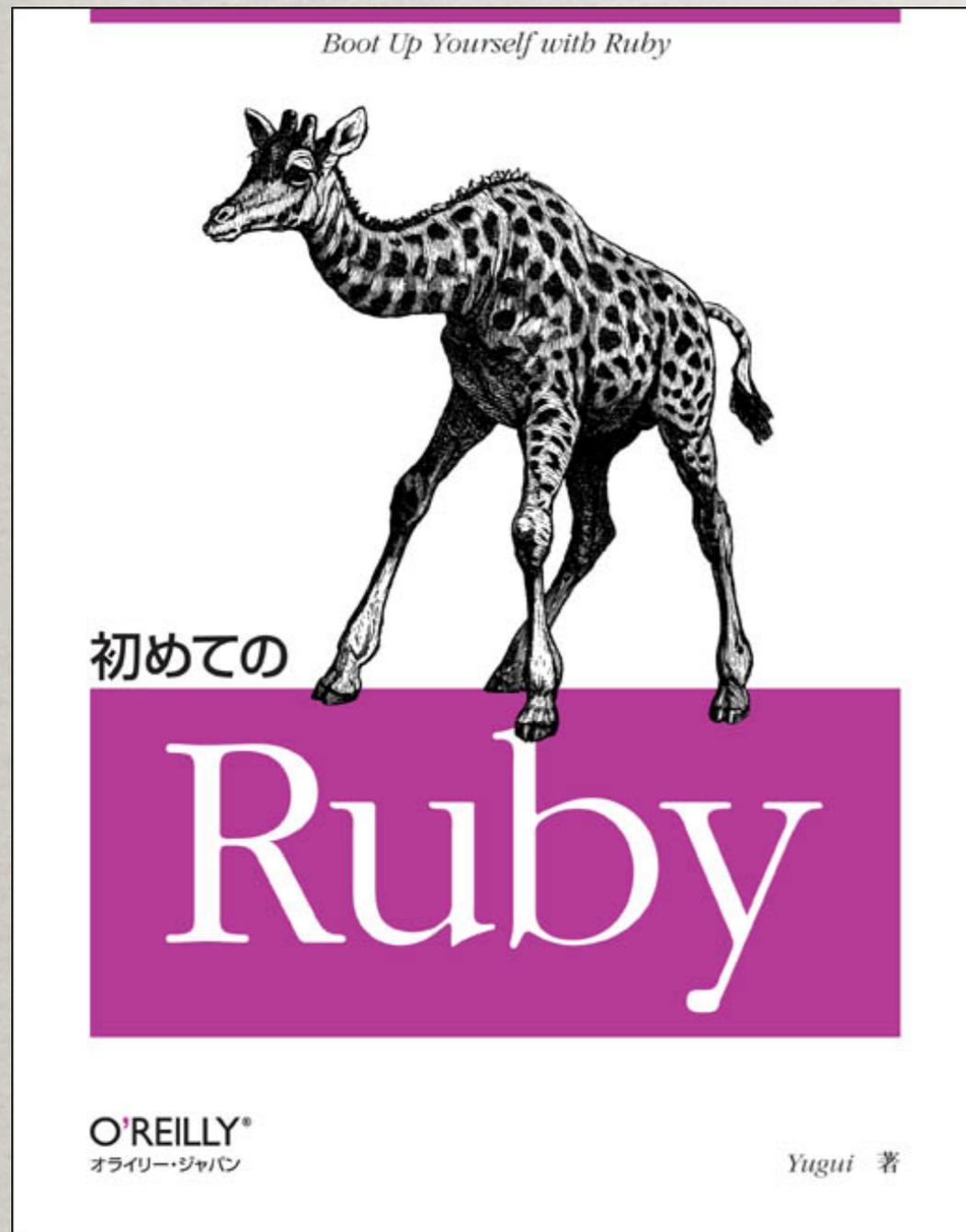
自己紹介

- ✻ Yugui (園田 裕貴)
- ✻ <http://yugui.jp>
- ✻ Ruby Issue Tracking System管理者
- ✻ 『初めてのRuby』

Ruby ITS

- ✻ Redmineを採用
- ✻ 独自拡張あり (Githubで公開)
- ✻ <http://redmine.ruby-lang.org>

初めてのRuby



- ✻ オライリー・ジャパン
- ✻ 224ページ
- ✻ 定価2,310円
- ✻ 多目的ホールで先行販売中

宣傳司了

では

Agenda

✻ 舞台

✻ 状況

✻ ツール

✻ お話

舞台

- ✻ 時に、2007年
- ✻ 今は亡き某社
- ✻ 公称30万PV/日の動画配信サイト
- ✻ 元はCOM+ASP.NET

舞台

✻ Tさん: 長老

✻ id:faultier

✻ 新人Yさん

✻ テスター/サポート Aさん, Bさん

状況

- ✻ 大幅に機能を追加予定
- ✻ 毎日バグが出る
- ✻ テストされていないコード
- ✻ 前回の機能追加は1ヶ月→3ヶ月

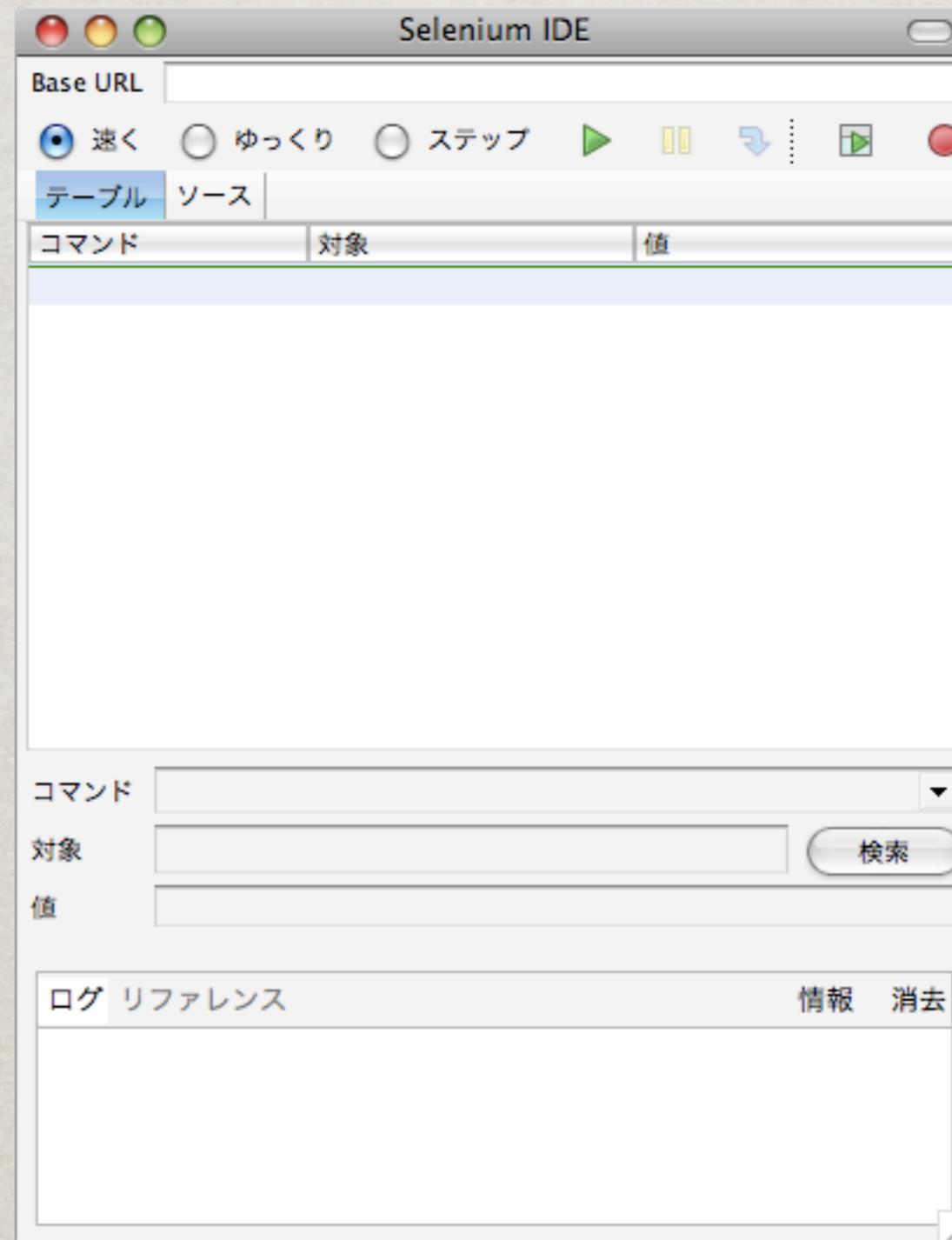
初期対応

- ✻ Subversion, Trac
- ✻ Build と Deploy の自動化
- ✻ モジュールごとの差し換え
- ✻ Selenium の導入

初期対応: 反応

- ✿ Subversion: 共有フォルダ代わりに
- ✿ Trac: すぐ慣れた
- ✿ Deploy自動化: 難航
- ✿ Selenium: Selenium IDEですぐに慣れた

Selenium IDE

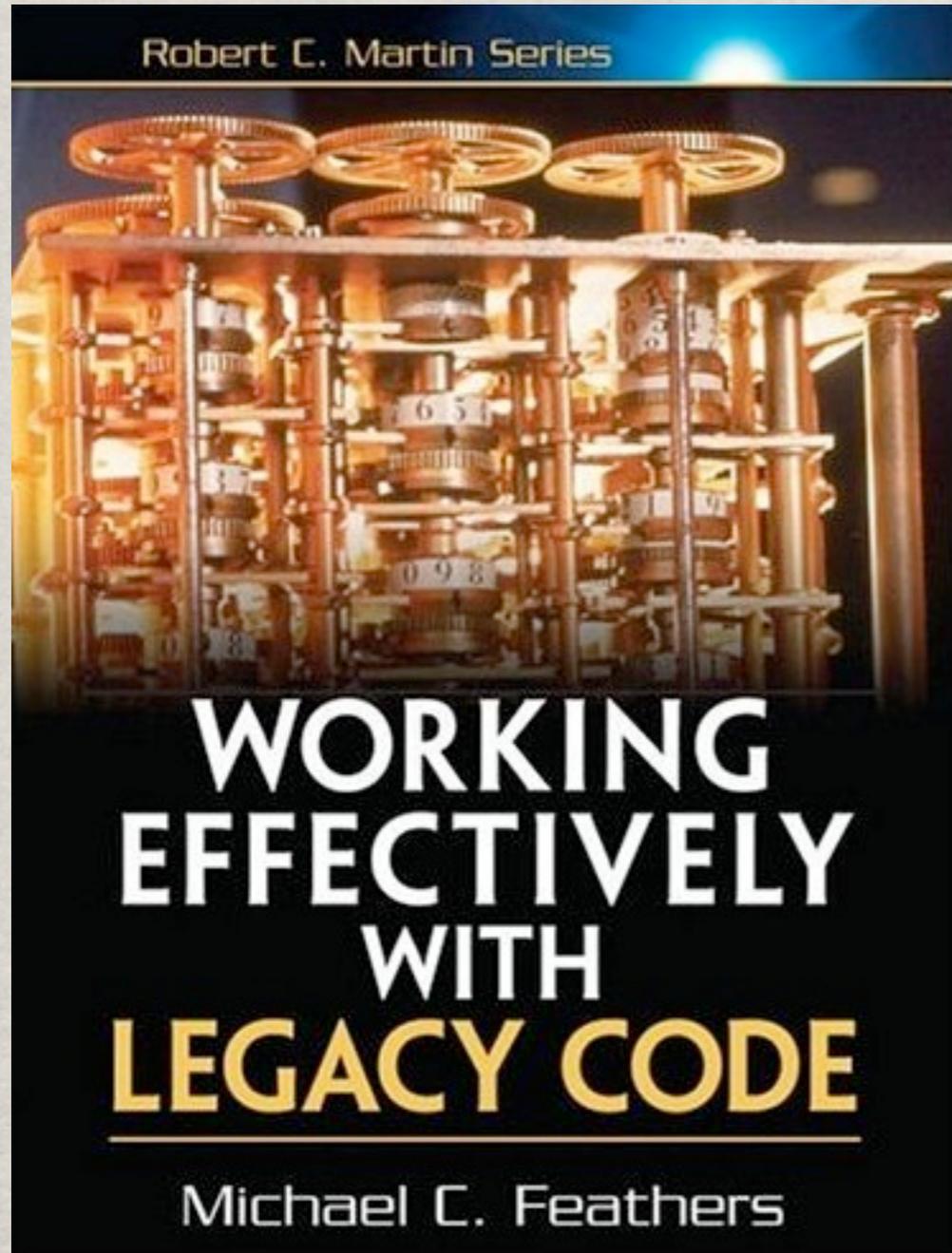


だかど

匙を投げる

- ✻ 事業方針上の加速要請
- ✻ まだ毎日出るバグ
- ✻ → 対応でベテランが1名必要
- ✻ 複雑怪奇な依存関係

WEwLC



Working Effectively with Legacy Code

要件

- ✻ 1ヶ月後のイベント
- ✻ 1ヶ月でリプレース + 機能追加
- ✻ サイト構成の試行錯誤への追隨

候補

✻ Seasar2

✻ Seasar2(.NET)

✻ Ruby on Rails

決定要因1

- ✻ そこにソースがあったから
- ✻ パンがないならパンを焼けばいいじゃない
- ✻ 文句があるならバザールにいらっしやい

決定要因2

- ✻ RSpec

- ✻ RSpec on Rails

- ✻ モック/スタブの充実

- ✻ 慣れてるし

結果

- ✻ phase 1 - リプレース - テストケース
- ✻ phase 2 - Specの徹底 - 普及
- ✻ phase 3 - BDD - 実践

phase 1

- ✻ Selenium資産の活用
- ✻ id:faultierとペアプロ
- ✻ Coverage低め
- ✻ 1ヶ月でリプレース完了

SeleniumRC Spec

✻ Motivation

- ✻ Seleniumを使いたいがGUIは嫌
- ✻ Selenium on Railsはなんか違う
- ✻ RSpecの表現力が欲しい

SeleniumRC =

- ✻ リモートプログラムから
- ✻ 多様な言語で
- ✻ ブラウザを操作
- ✻ RSpecからも簡単

SeleniumRC Spec

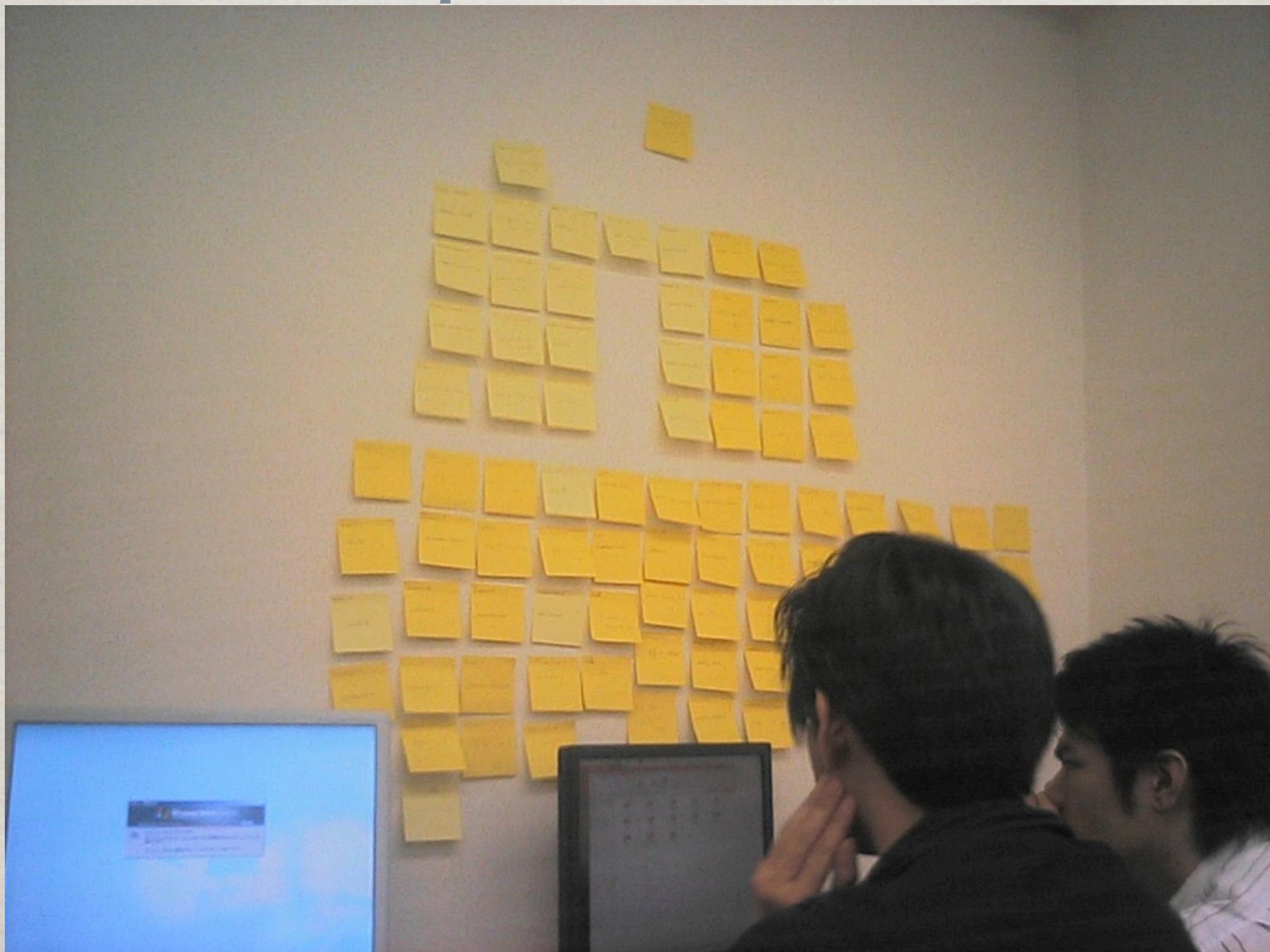
```
require File.dirname(__FILE__) + '/../spec_helper'  
  
story "A reader visits our site" do  
  scenario "The reader", "views our toppage" do  
    # You can use "he", "she" or "ve"  
    she.opens "/"  
    she.gets_title; it.should == "Hello, World!"  
  end  
end
```

<http://coderepos.org/share/wiki/SeleniumRcSpec>

phase 2

- ✻ Unspecified classesを端から処理
- ✻ Coverage 100%へ
- ✻ コードの7割はSpecに

phase 2



phase 2

- ✻ 「どういう風にSpecを書けばいいかわからない」
- ✻ ペアプロによる勇気
- ✻ 適宜ペアを組み替え
- ✻ Conventionによる強制

phase 3

- ✻ 厳密なBDDへ
 - ✻ 機能決定
 - ✻ スケジュール
 - ✻ Selenium
 - ✻ models, controllers, views, helpers

phase3: 機能決定

The whiteboard contains the following handwritten notes and diagrams:

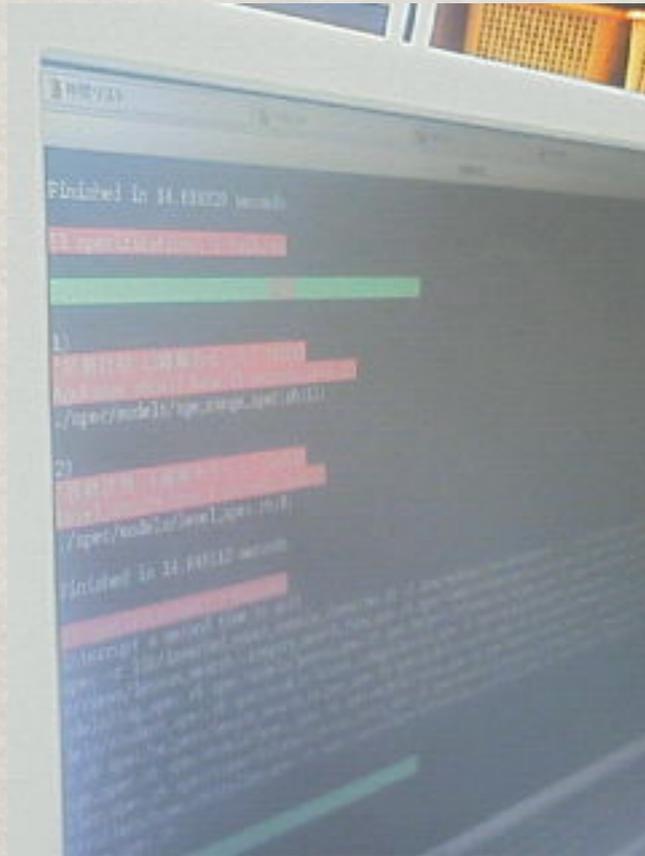
- login系** (login system) with a circled sticky note and the word **lesson** written below it.
- community** with a circled sticky note and the word **lesson** written below it.
- logging** written vertically on the right side.
- green化** (greening) circled in red.
- logging of bug fix** and **予知と改善 improve** (prediction and improvement).
- deploy** circled in red.
- 2) 動画CMを作る? & mate.** (2) Create video CM? & mate.
- 1/4** circled in red.
- 2)** circled in red.
- オリジナル(2-4-1)の作りかた** (Original (2-4-1) creation method).
- かわいさ** (cuteness).
- 仕様** (specification).
- シマウサギ** (Shimazumi).
- lesson** written at the bottom left.

A small photo of a sloth is visible in the bottom right corner of the whiteboard area.

phase 3: スケジュール



phase 3: BDD



- ☼ Red
- ☼ Green
- ☼ Refactor

結果

- ✻ phase 1 - 先行実施
- ✻ phase 2 - 真似ながら試行
- ✻ phase 3 - 消化完了

結果

- ✻ BDDの浸透
- ✻ エラーはnilが問題
 - ✻ 動的型付けのリスクは排除
- ✻ 「Specがあると安心できる」

おしましい

ご静聴

ありがとうございました