

クは駆動のク

“D is for Driven”

～Developer TestingとBDD～

(株)永和システムマネジメント

角谷 信太郎

s-kakutani@esm.co.jp

オブジェクト倶楽部
2006クリスマスイベント

技術トラック

2006-12-20(水)

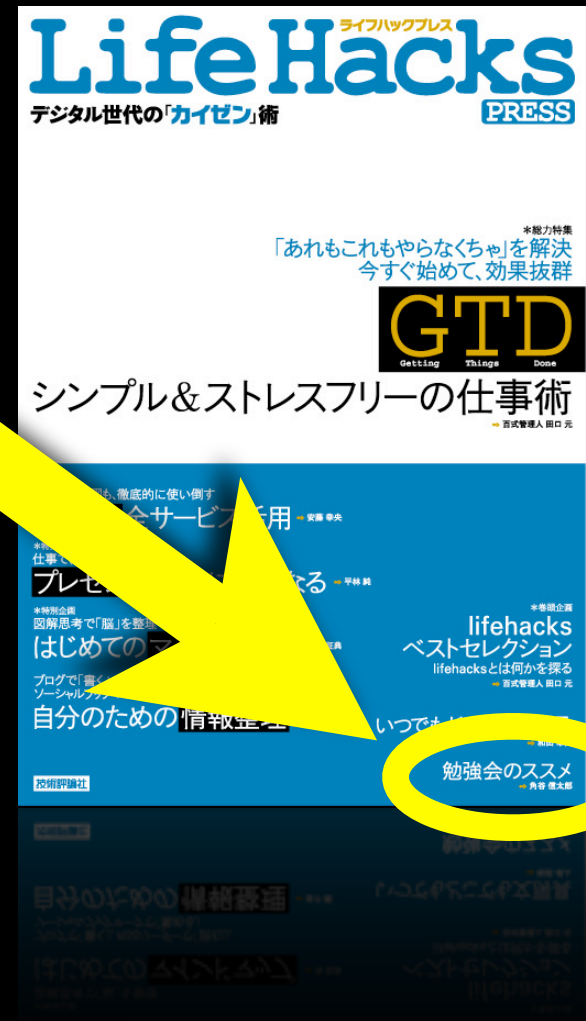
「——ぼくは心を相手に戦いを始めた。
からだが生産できないものを、心がほし
がるからさ」

レイ・ブラッドベリ 『ウは宇宙船のウ』

自己紹介

角谷 信太郎

- (株)永和システムマネジメント
- **テスト駆動**開発者
- 日本**Ruby**の会(理事)
- 最近停滞中
- Quick-JUnit Plugin
- 勝手に翻訳
- ときどき執筆



宣 行

23

(T)

『月刊JavaWorld』 の次号(休刊号)

(2)

デブサミ2007



デブサミ2007

Developers
Summit

デブサミ2007

- 時: 2007-02-14(水), 15(木)
- 於: 目黒雅叙園(東京・目黒)
- 主催: 翔泳社
- <http://seshop.com/event/dev/2007/>

実践 『From Java To Ruby』 ～血があつい

鉄道ならば/走りぬけて

ゆく自動車はいつかは心

臓を通るだるう～

謝辞

- t-wadaさん

- 私のTDDコーチ

- 懸田 剛さん

- 私にBDDを教えてくださいました

謝辞(順不同)

水越さん、伊藤コウイチさん、
id:secondlifeさん、平鍋さん、
諸橋さん、須藤さん、天野さん、
Yuguiさん、宮川さん

よろしく

お願いいたします

Developer Testingと BDDについて

Developer Testing

Developer Testing

- **Developer Tests**

- 開発者が行う、開発促進のためのテスト
- QA Tests

- **Customer Tests**

- 顧客視点からのシステム機能のテスト

Developer Testing

- Developer Tests

- 開発者が行う、開発促進のためのテスト

- QA Tests

- Customer Tests

- 顧客視点からのシステム機能のテスト

BDD

ビヘイビア

振舞駆動開発

Behaviour Driven Development

“**BDD** is the natural
evolution of **TDD**.”

——Steven R.Baker

<http://groups.yahoo.com/group/extremeprogramming/message/113518>

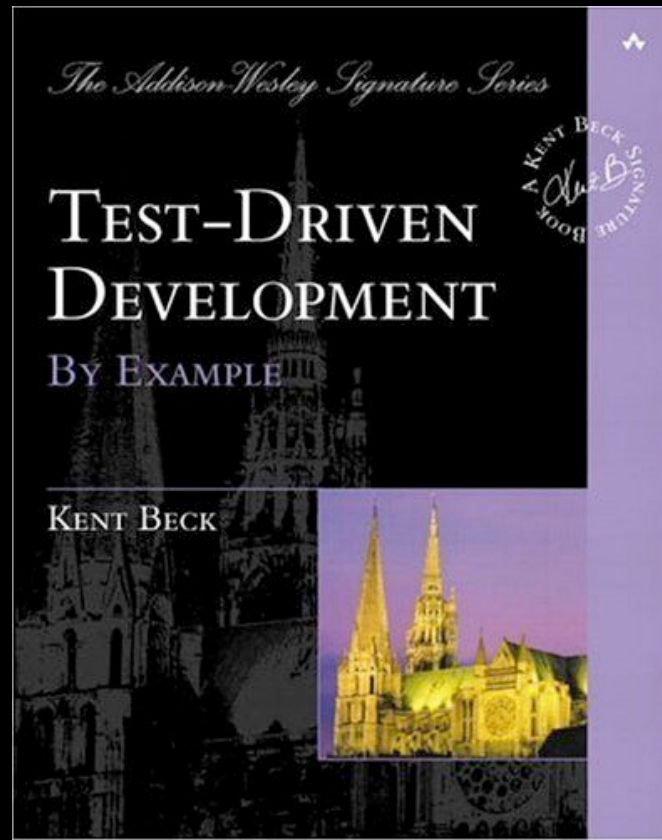
BDDはTDDの正常進化である

TDD

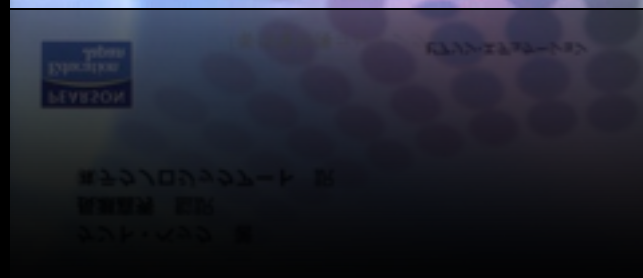
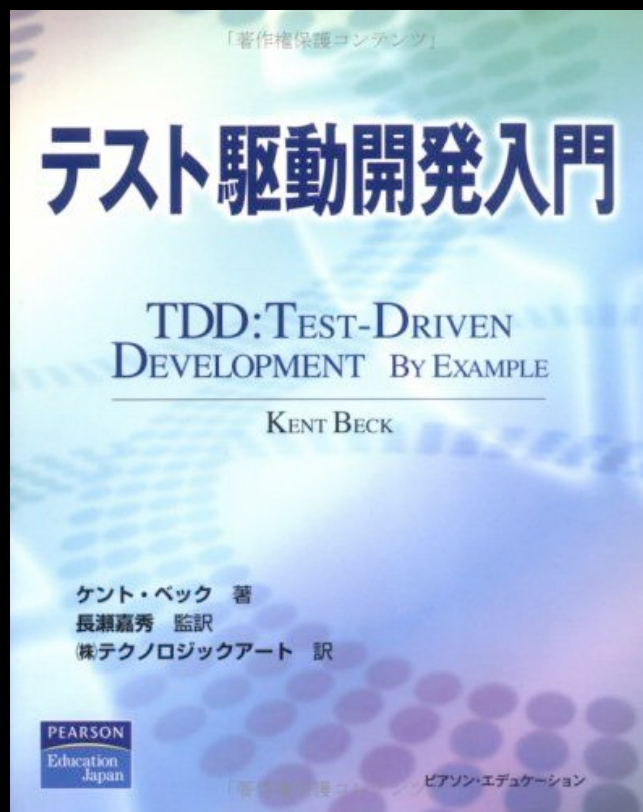
テスト駆動開発

Test Driven Development

TDD by Example



テスト駆動開発入門



偉大な書籍は
偉大な1行から
はじまる

Clean code that works, in
Ron Jeffries' pithy phrase,
is the goal of Test-Driven
Development(TDD).

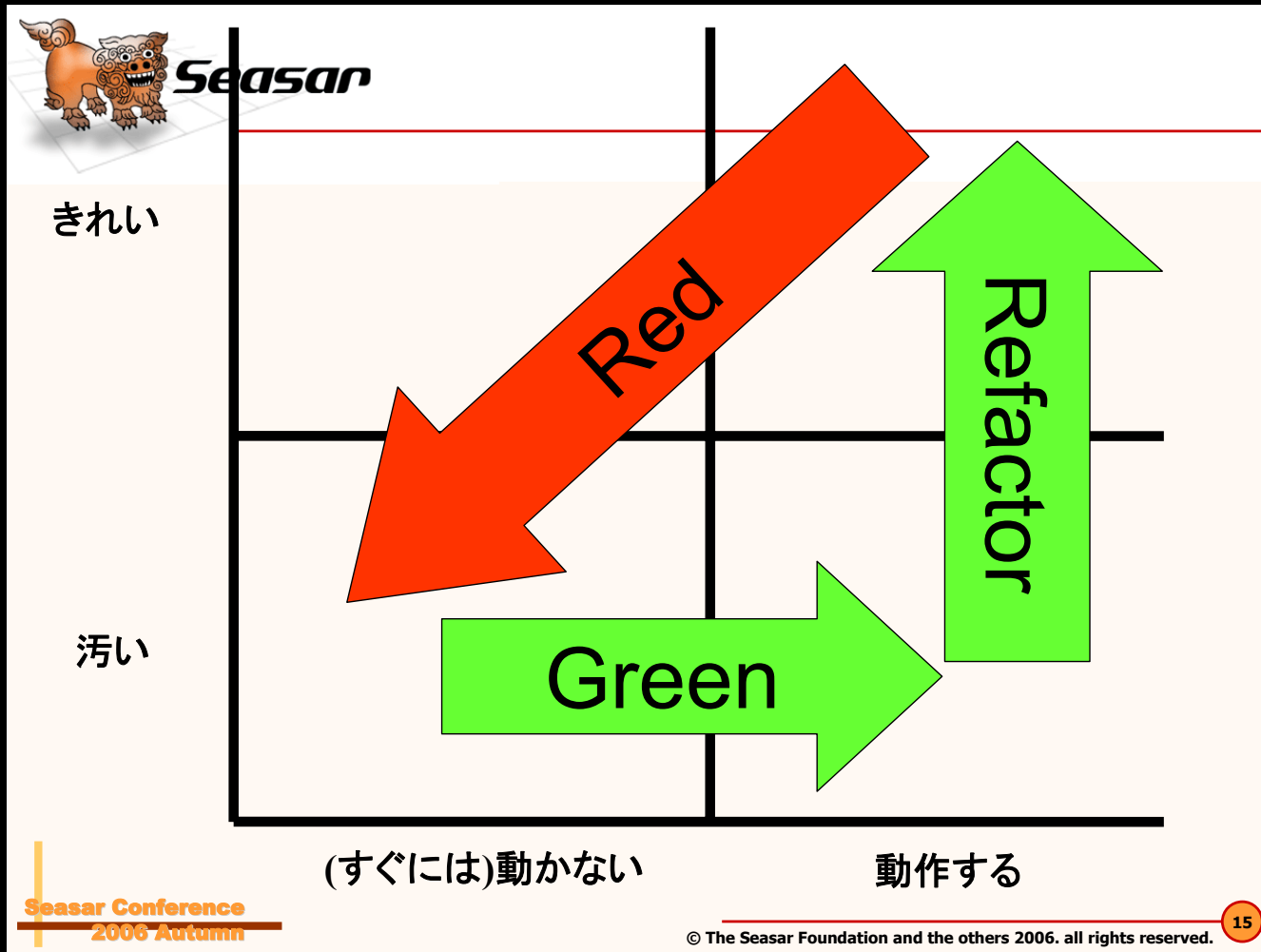
「動作するきれいなコード」、
このRon Jefferiesの簡潔な言葉こそが
TDDのゴールである。

動作する

きれいなコード

“Clean code that works”

動作するきれいなコードとR-G-R



“The translation of a feeling into a test is a common theme of TDD.”

感情をテストにすることが、TDDに共通するテーマである。

感情

- **不安**: 自分は仕事は正しいか?
- **自信**: 自分の仕事は正しい!
- **勇気**: 不安を克服する意志
- **信頼**: 自分が仲間を、仲間が自分を
- **退屈**: このテストは必要か?

馬車重動

駆動

- 「動作するきれいなコード」への意志
- 「設計」とフィードバックとのギャップ認識/ギャップ制御
- 感情

『WEB+DB PRESS』

vol.35



t-wadaさんによる

TDDの

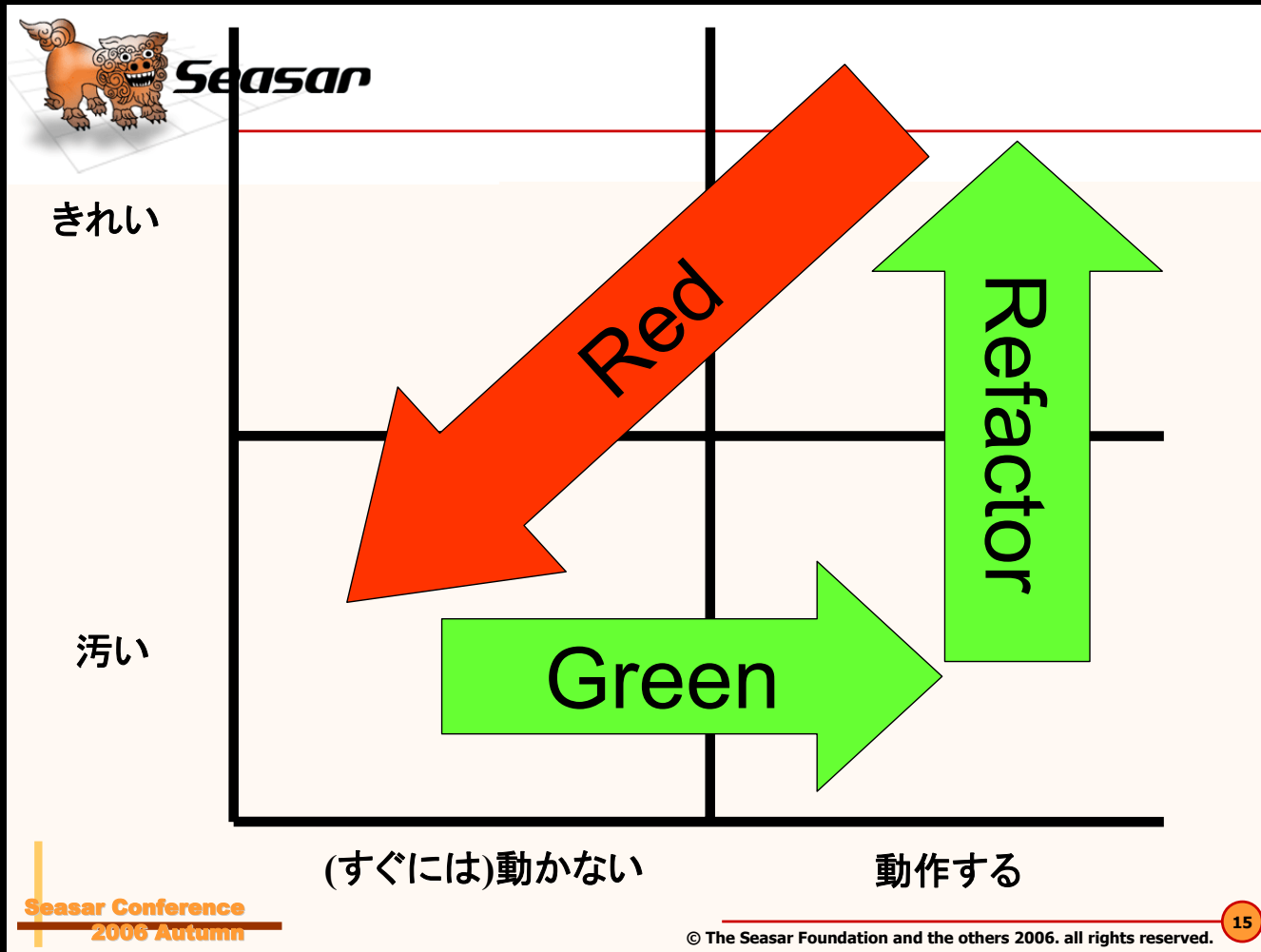
思想・技術・実践

心技体

t-wada says...

- プログラマのための**設計技法**
- **R-G-R**のリズムを体得せよ
- **心理**マネジメント手法
- 無駄なく**健康**なコード
- テストコードの**資産価値**

Red-Green-Refactor



Kent Beckも言ってる:

- プログラマのための**設計技法**
- **R-G-R**のリズムを体得せよ
- **心理**マネジメント手法
- 無駄なく健康なコード
- テストコードの資産価値

t-wadaが言っていたよ:

- プログラマのための設計技法
- R-G-Rのリズムを体得せよ
- 心理マネジメント手法
- 無駄なく**健康**なコード
- テストコードの**資産価値**

Kent Beck とBDD

- > So, stop thinking of TDD and
- > BDD as conflicting ideals.
- > **BDD** is the natural **evolution** of
- > **TDD**.

What evolved?

——Kent Beck

<http://tech.groups.yahoo.com/group/extremeprogramming/message/113575>

何が進化したって？

アジェンダ

- TDDからBDDへの”進化”によって:
 - 何が**変わる**のか?
 - **変わらない**ものはあるのか?
 - 変わらないものがあるならば
それは何なのか?

ま と め

BDDとは:

- 顧客と同じ**語彙**を使って記述したシステムの**振る舞い**からの**フィードバック**と、
- 開発者の**感情**によって、
- **駆動**される開発

まとめ

- BDDはTDDのマインドセットを**より明確にする**
- BDDはマインドセットを明確にする**語彙とツール**を提供する
- BDDもTDDも、**具体的なフィードバック**と開発者の感情が開発を**駆動**する

もくじ

- BDDへの道
- BDDで**変わる**もの
- BDDで**変わらない**もの
- RSpecでBDDのデモ

BDDへの道

稻垣足穂

詩は歴史に

対して

垂直に立つ

実存としてのBDDer

- 人は、
 - BDDerとして「**在る**」のではない
 - BDDerに「**成る**」のだ
 - どうやって「成る」のか？

止此喻

個体発生は
系統発生を
繰り返す

反復説

ある動物の発生
の過程は、その
動物の進化の過
程を反復する。



2006:

a **testing** odyssey

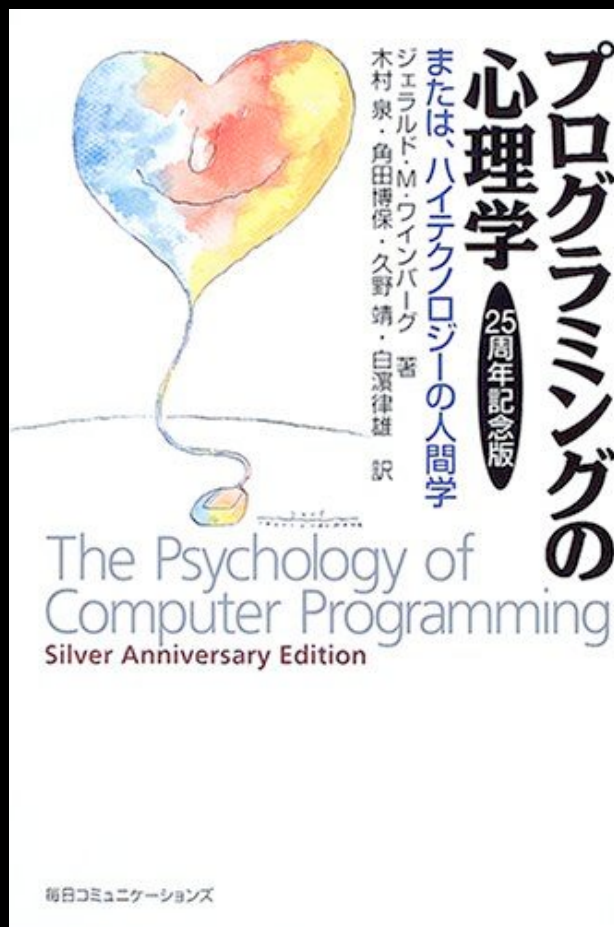


テストイングの夜明け

The Dawn of The Testing

1971

プログラミングの心理学



ジェラルド・M・ワインバーグ
(初版は1971年)

(プログラマがテストしなくなる)
この過ちから身を守るための一つ
の方法は、**テスト問題をテストを**
はじめる前に、いやそれどころか
もし可能ならば**コーディング前に**
作っておく、というものである。

1992

ソフトウェア 設計とは何か？

What is Software Design?

By Jack W. Reeves

村上 雅章さんによる翻訳

ソフトウェア工学とは何か

ソフトウェア設計とは何か？

(原文： [What Is Software Design?](#))

by

Jack W. Reeves
(c)C++ Journal - 1992

[簡単マニュアル作成ツール](#) [テスト管理ソリューション](#)
マニュアル作成コストを大幅 システムの品質を本質的に向上
カット さらに対話型マニュアルをさせる 品質管理プロセスの自動

0052699

訳者まえがき

この文書は、Jack W. Reeves 氏が1992年に C++ Journal に寄稿した記事の邦訳です。本記事では、オブジェクト指向プログラミング言語の代表として C++ を挙げていますが、これは本記事が執筆された当時、一般的に利用可能なオブジェクト指向言語は C++ だけであったという事情があるためです。今では C++ に加えて Java, Delphi, C# といったオブジェクト指向言語が利用可能となっていますが、そんな今でさえこの記事は古さを感じないものとなっており、ソフトウェア開発の本質、現状を鋭くえぐるものとなっています。

邦訳の公開を許諾していただいた Jack W. Reeves 氏に、この場を借りて感謝いたします。

2003年1月27日 村上 雅章

5003年1月27日 村上 雅章

邦訳の公開を許諾していただいた Jack W. Reeves 氏に、この場を借りて感謝いたします。

邦訳は早急で公開させていただきます。入稿の件は別途お知らせいたします。

<http://www.biwa.ne.jp/~mmura/SoftwareDevelopment/WhatIsSoftwareDesignJ.html>

ソフトウェア設計とは何か？

- 設計: ソースコード
- 製造: ビルド

コーディングは設計であり、テスト
とデバッグも設計の
一部であり、私たちが一般的にソ
フトウェア設計と呼んでいるもの
もやはり設計の一部なのです。

ソフトウェア設計はコーディング
が完了し、「かつ」テストされる
までは完璧にならないのです。そ
して。テストは設計の検証
と洗練を行うプロセスにおける礎
となるものです。

A pixelated, low-resolution image of a sunset or sunrise. The sun is a bright, glowing square in the center, partially obscured by a dark, jagged horizon line. The sky is filled with various shades of brown, tan, and orange, creating a textured, mosaic-like effect.

太初にテストファーストありき

In the beginning was the test-first

1998

JUnunit1.0

テスト熱中症

プログラマは、テストを書く
のが好きになる

“Test Infected: Programmers Love
Writing Tests”

By Eric Gamma

小野 剛さんの翻訳

The screenshot shows a web browser window with the title "- テスト熱中症". The page header includes the "Object Club" logo and navigation links: ホーム, 技術文書, コミュニティ, スペシャル, ダウンロード, イベント. A search bar and the text "Powered by Eiwa System Management, Inc. 個人情報の取扱について" are also visible.

The main content area displays the article title: **JUnit テスト熱中症：プログラマは、テストを書くのが好きになる**. Below the title is the original source: **原題: Test Infected: Programmers Love Writing Tests**.

The article text begins with: "テスト作業が、開発作業の中にしっかりと組み込まれていない。こうなると、開発の進み具合を計測することは不可能になってしまう。というのも、あるの機能が動き始めたのはいつか、またある機能が動かなくなったのはいつか、まったくわからなくなるからだ。JUnit を使えば、苦勞なく、しかも段階的に、テストスイートを構築できる。このテストスイートは、進捗状況を把握したり、意図しない副次効果を見つけたしたり、また開発で労力をかけるべき箇所を明らかにしたりする上で役に立つだろう。"

A table of contents follows, listing sections: 問題, 実例, テストする習慣 (プラクティス), and 結語.

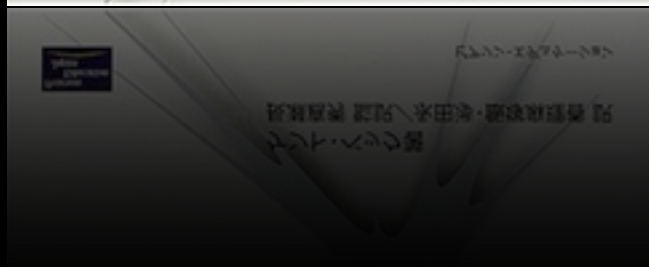
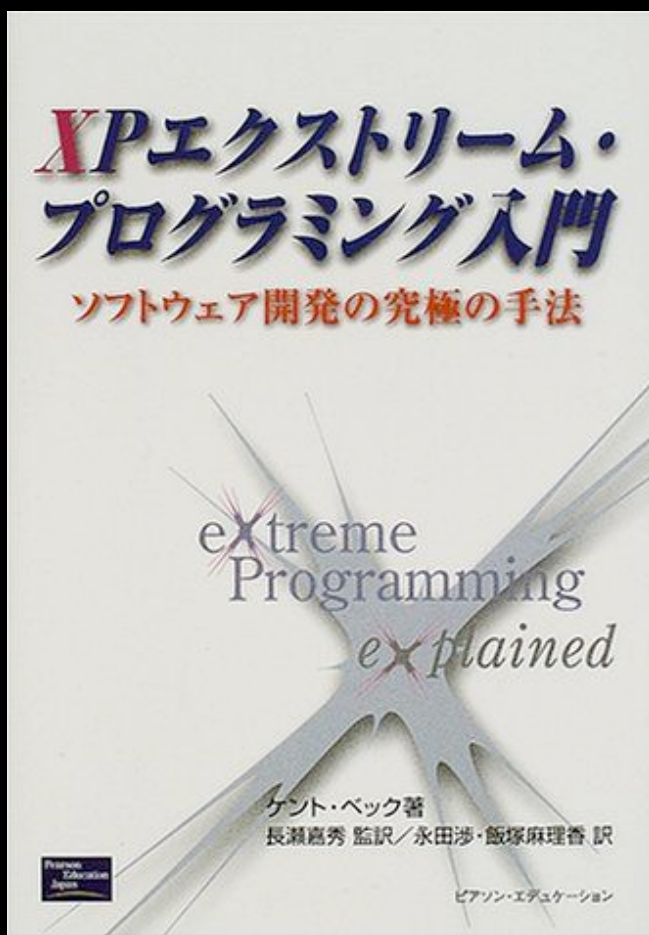
The "問題" section starts with: "どんなプログラマだって「コードを書いたらテストも書くべきだ」ということは知っている。にもかかわらず、実際に書いている人は少ない。「何で書かなかったの?」という質問に対しては、「あまりにも忙しすぎて」という同じ 答えが返ってくる。しかしこれは悪循環

http://www.objectclub.jp/community/XP-jp/XP_relate/testinfected-j

システムはこう動くはずだ、
という考えが頭に浮かんだと
しても、実装について考える
のは後回しにしよう。まずは
テストを書くのだ。そしてテ
ストを実行するのだ。

1999

XP:explained

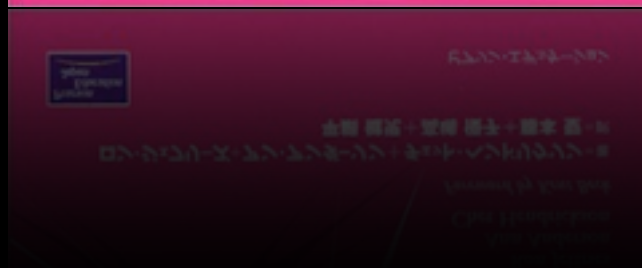
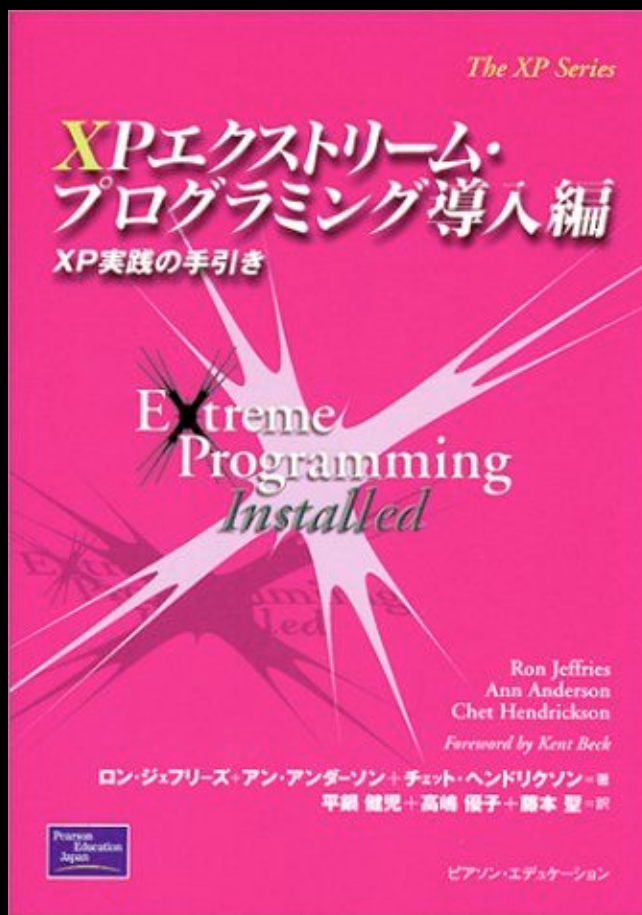


テストファースト
という単語自体は
出てこない
(意外)

XPのテストは...プログラムを
動作させることが好きな人の
仕事である。...テストはあくま
でも道具であって、本来重要
なものはシステムの振る舞いで
ある。

20000

XP:installed



テストファースト
という単語が
広まるきっかけ？

テストファーストの効果

- テスト可能な設計
- コード利用者の視点
- インターフェイス中心の設計

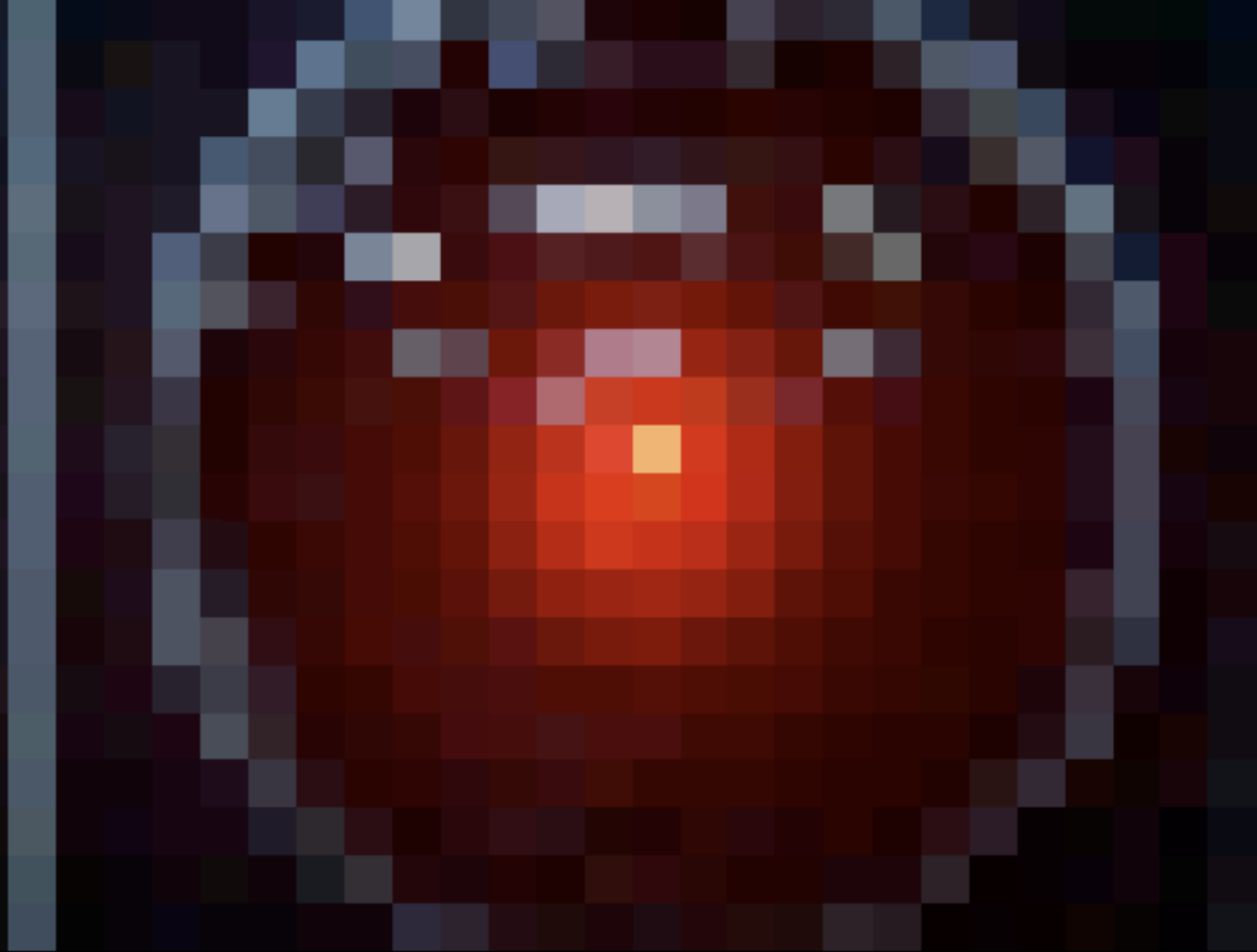
『WEB+DB PRESS Vol.35』のt-wada記事から要旨抜粋

どうみても

設計です

コーディングは設計であり、テスト
とデバッグも設計の
一部であり、私たちが一般的にソ
フトウェア設計と呼んでいるもの
もやはり設計の一部なのです。

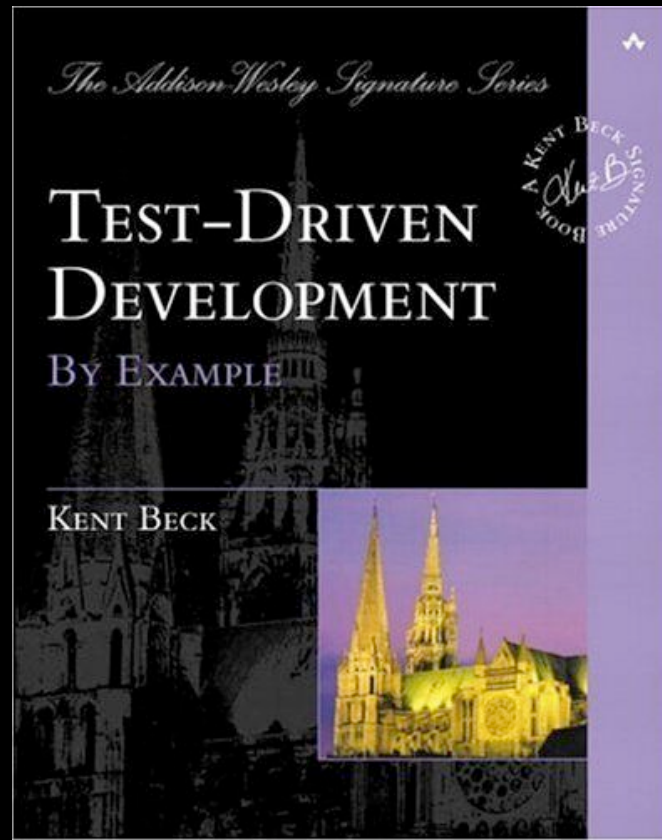
2002



テスト駆動開発

Test Driven Development

TDD by Example



t-wada says...

- プログラマのための**設計技法**
- **R-G-R**のリズムを体得せよ
- **心理**マネジメント手法
- 無駄なく**健康**なコード
- テストコードの**資産価値**

2003



ビヘイビア

振舞駆動開発

Behaviour Driven Development

BDDの誕生

- Dan Northが
JBehaveプロジェクトを開始

20004

日本への輸入

- ADC2004にて、
平鍋さんがJBehaveに遭遇
- Agile Development Conference
2004
- 廊下でデモしていたらしい
 - http://blogs.itmedia.co.jp/hiranabe/2005/10/tdd_bdd_731d.html

2005

テスト駆動開発について の新しい考えかた

“A New Look at Test Driven
Development”

By Dave Astels

懸田 剛さんによる翻訳

BDDIntro - 生きてま2 改 - Trac

生きてま2

Search

[Login](#) | [Settings](#) | [Help/Guide](#) | [About Trac](#)

[Wiki](#) | [Timeline](#) | [Search](#) | [WaistreelMemo](#) | [Tags](#) | [Blog](#)

[Start Page](#) | [Title Index](#) | [Recent Changes](#) | [Page History](#)

A NEW LOOK AT TEST-DRIVEN DEVELOPMENT

Author: Dave Astels <dastels_at_daveastels.com>
Original: <http://daveastels.com/files/sdbp2005/BDD%20Intro.pdf>
Translator: Takeshi Kakeda <t.kakeda_at_gmail.com>
Reviewer: Shintaro Kakutani (<http://kakutani.com/>), nishikawa (<http://d.hatena.ne.jp/nskj77/>)
Updated: 2005/12/20

Table of Contents

- [1 はじめに\(日本語版のみ\)](#)
- [2 問題](#)
 - [2.1 テスティングが焦点](#)
- [3 結果](#)
- [4 テストじゃないなら何なのか?](#)
- [5 サビア-ウォーフの仮説](#)
- [6 ではどうすると?](#)
- [7 ビヘイビア仕様フレームワーク](#)
- [8 今は何があるの?](#)
- [9 ガイドライン](#)
- [10 まとめ](#)
 - [10.1 謝辞](#)

1 はじめに(日本語版のみ)

Behaviour Driven Developmentについての初の記事ですが、訳語について悩んでいます。現在は以下のようにしてあります。

<http://giantech.jp/wiki/BDDIntro>

BDDが認知される契機に

- TDDで定義しているのは、テストというより**振る舞い**では？
- xUnitはテストの**語彙**と、プログラミング**言語の構造**に囚われすぎでは？
- 新しい**考え方**には新しい**ツール**

新ししい

ツール

RSpec

**Framework
for Behaviour
Driven
Development
in Ruby**

初期の

RSpec

```
class EmptyMovieList < Spec::Context
  def setup
    @list = MovieList.new
  end

  def should_have_size_of_0
    @list.size.should_equal 0
  end

  def should_not_include_star_wars
    @list.should_not_include "Star Wars"
  end
end
```

```
class OneMovieList < Spec::Context
  def setup
    ...
  def should_have_size_of_1
    ...
  end
end
```

20006

2006

- **RSpec on Rails** => **利用者増加**
- Introducing Behaviour-Driven Development
<http://www.dannorth.net/introducing-bdd> (by Dan North)
- “**Given, When, Then**”の提唱
- 他言語によるBDD実装
- 本邦Ruby界(の一部)でも**ブレイク寸前?**

RSpec on Rails

割愛

“Given, When, Then”

割愛

他言語によるBDD実装

- **.NET**: NSpec, NSpecify
- **Java**: JBehave, (rMock)
- (**Ruby**: RBehave)

C#が健闘中!

本邦Ruby界(の一部)でもブレイク寸前?

- 池上さん
 - KOF2006、Ruby@関西
- 諸橋さん
 - Rails勉強会(東京)

いまこそ

2006: a testing odyssey

- ソースコードとテストは**設計**
- **xUnit**, test-first
- **TDD**
- BDD, **RSpec**

**BDDで変わるもの/
変わらないもの**

ビヘイビア

振舞駆動開発

Behaviour Driven Development

BDD is...

- an **evolution**
- **in the thinking** behind:
 - Test Driven Development
 - Acceptance Test Driven Planning

<http://behaviour-driven.org/>

BDD は...

- 進化した(evolution)
- 考え方(thinking):
 - テスト駆動開発
 - 受入テスト駆動プランニング

<http://behaviour-driven.org/>

BDDのゴール

- Developer Testingと、Customer Testingの**統合**
- システムの**振る舞い**を定義する
- “Getting the Words Right”
 - 「正しい言葉を選ぶ」
⇒ 顧客と同じ**語彙**で設計する

BDDはゴールに向けて

- **マインドセット**を変える
- **語彙**を変える
- **ツール**を変える

BDDで変わるもの

BDDで変わるもの

1. マインドセット

2. 語彙

3. ツール

(T)

マインドセット
を変える

マインドセット

- TDDっていうけど、
- 開発を駆動してるこれって、
- テストではなく、
- システムの振る舞いでは？

マインドセット

- 「テスト」と呼ぶのは**混乱**の元
- 「振る舞い」は「振る舞い」だ
- **開発**を**駆動**するのは**振る舞い**だ

ビヘイビア

振舞駆動開発

Behaviour Driven Development

(私見)

振舞

ふるまい

ビヘイビア

発音し

づらい

振舞は
テストで
はなない？

Test (by Kent Beck)

【動詞】 評価する

【名詞】 受入または拒否を
導く手続き

『テスト駆動開発入門』 第25章

どうみても

テストです

テストは

発音しやすい

(2)

語彙を

変える

サピアニウオーフ仮説

Sapir–Whorf hypothesis

言語は我々の思考を形成し、
思考できる内容を決定する。

ベンジャミン・ウォーフ

語彙:TDD

- **TestCase**

- テストケース。クラス。

- **Test Method**

- テストメソッド。メソッド。

- **assert**

- アサート。結果の確認。

テストに
偏りすぎ

語彙:BDD

- Context

- コンテキスト。No TestCase!

- specify

- 「仕様」の定義。No test_method!

- expectation

- システムに期待する振る舞い。No assert!

(私見)

スペシフアイ

エクスペク

テーション

やっぱり

発音

しづらしい

語彙:TDD

• Unit Test

- 「単位」って? クラス? メソッド?
- “small-scale tests” -- Kent Beck
- 製品コードと対応するの? しないの?

プログラミング

言語の構造に

偏りすぎ

語彙:BDD

- spec file

- 振る舞いの論理的なまとめ
- RSpecではファイル
- 1ファイルにつき、nコンテキスト

calc_spec.rb:

```
context "1+1の足し算" do
  setup do
    @sum = 1 + 1
  end

  specify "合計値は2であること" do
    @sum.should_eq 2
  end
end

context "引き算" do
  specify "2-0は0であること" do
    (2 - 0).should_eq 0
  end
end
```

expectation

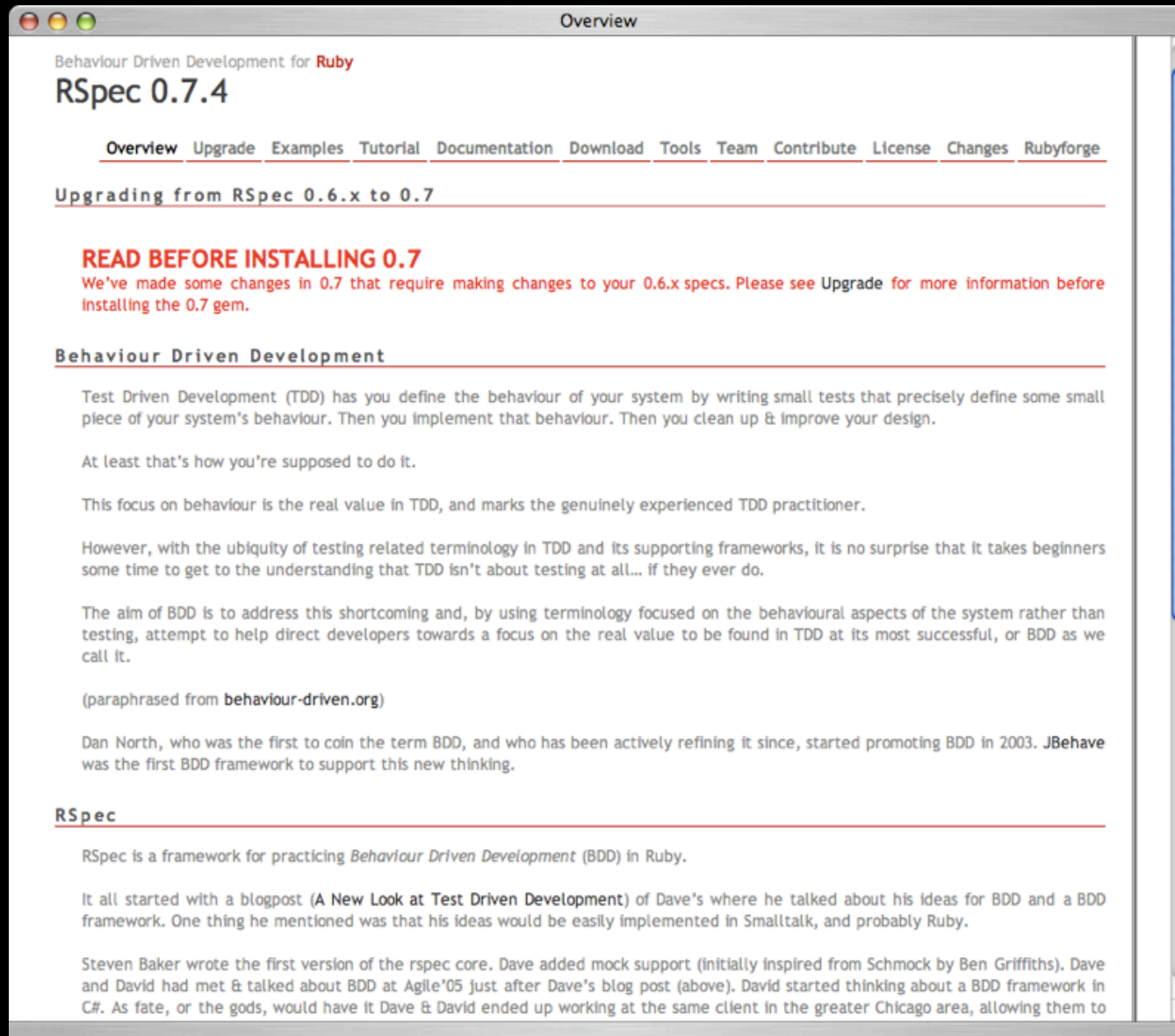
(3)

ツールを
変える

RSpec

**Framework
for Behaviour
Driven
Development
in Ruby**

rspec.rubyforge.org



Overview

Behaviour Driven Development for **Ruby**
RSpec 0.7.4

[Overview](#) [Upgrade](#) [Examples](#) [Tutorial](#) [Documentation](#) [Download](#) [Tools](#) [Team](#) [Contribute](#) [License](#) [Changes](#) [Rubyforge](#)

Upgrading from RSpec 0.6.x to 0.7

READ BEFORE INSTALLING 0.7
We've made some changes in 0.7 that require making changes to your 0.6.x specs. Please see [Upgrade](#) for more information before installing the 0.7 gem.

Behaviour Driven Development

Test Driven Development (TDD) has you define the behaviour of your system by writing small tests that precisely define some small piece of your system's behaviour. Then you implement that behaviour. Then you clean up & improve your design.

At least that's how you're supposed to do it.

This focus on behaviour is the real value in TDD, and marks the genuinely experienced TDD practitioner.

However, with the ubiquity of testing related terminology in TDD and its supporting frameworks, it is no surprise that it takes beginners some time to get to the understanding that TDD isn't about testing at all... if they ever do.

The aim of BDD is to address this shortcoming and, by using terminology focused on the behavioural aspects of the system rather than testing, attempt to help direct developers towards a focus on the real value to be found in TDD at its most successful, or BDD as we call it.

(paraphrased from [behaviour-driven.org](#))

Dan North, who was the first to coin the term BDD, and who has been actively refining it since, started promoting BDD in 2003. [JBehave](#) was the first BDD framework to support this new thinking.

RSpec

RSpec is a framework for practicing *Behaviour Driven Development* (BDD) in Ruby.

It all started with a blogpost ([A New Look at Test Driven Development](#)) of Dave's where he talked about his ideas for BDD and a BDD framework. One thing he mentioned was that his ideas would be easily implemented in Smalltalk, and probably Ruby.

Steven Baker wrote the first version of the rspec core. Dave added mock support (initially inspired from Schmock by Ben Griffiths). Dave and David had met & talked about BDD at Agile'05 just after Dave's blog post (above). David started thinking about a BDD framework in C#. As fate, or the gods, would have it Dave & David ended up working at the same client in the greater Chicago area, allowing them to

絶賛

開発中

APIや内部

設計の変更

が激しい

紹介する

けど使わな

くてもいい

私とRSpec



Team

Behaviour Driven Development for **Ruby**
RSpec 0.7.4

[Overview](#) [Upgrade](#) [Examples](#) [Tutorial](#) [Documentation](#) [Download](#) [Tools](#) [Team](#) [Contribute](#) [License](#) [Changes](#) [Rubyforge](#)


The Core Team

- Dave Astels
- Steve Baker
- David Chelimsky
- Aslak Hellesoy
- Brian Takita

David and Aslak Dave Astels


Contributors



私とRSpec

Team
David and Astak Dave Astels

Contributors



RSpec

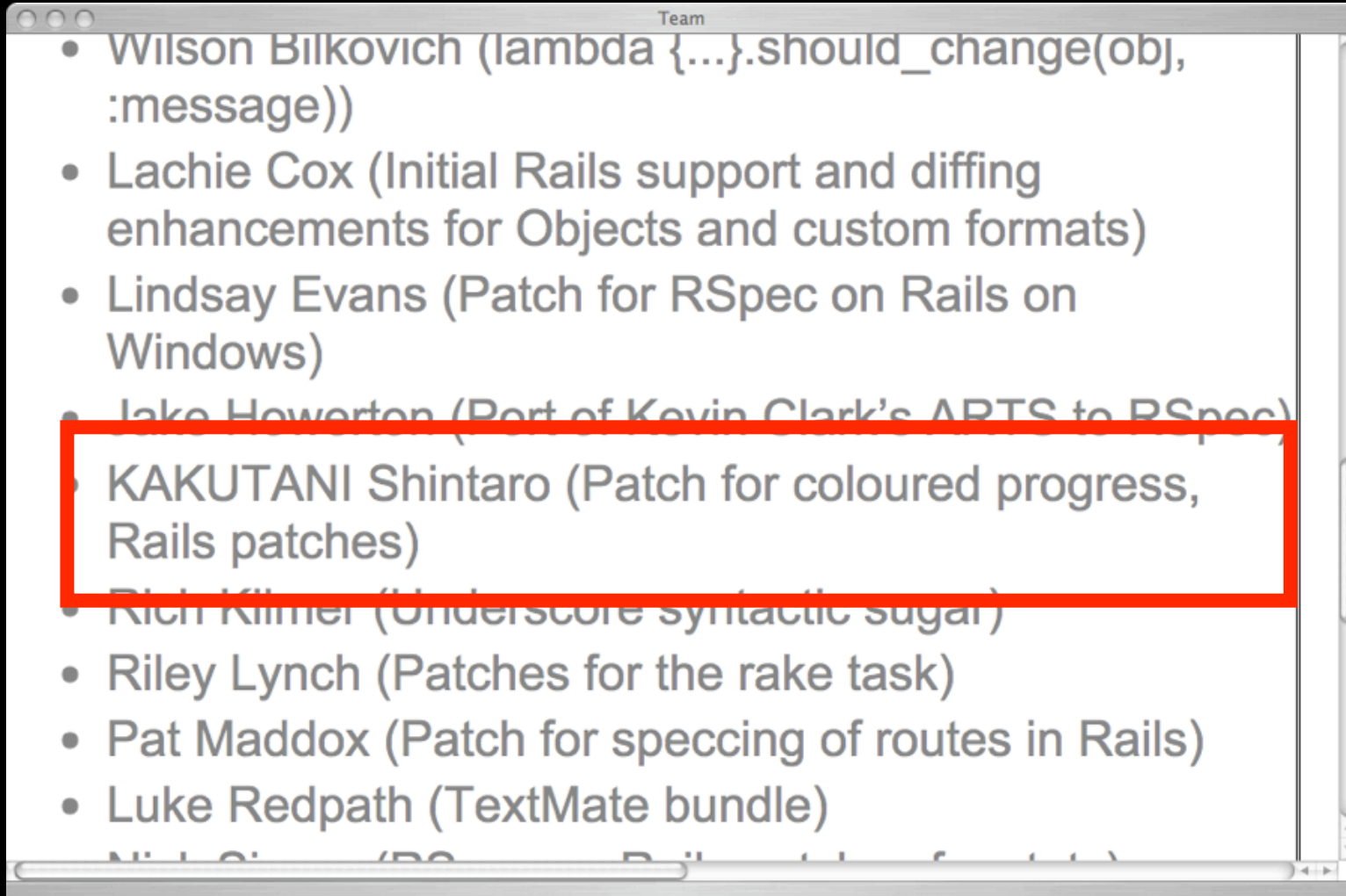
In alphabetical order (by last name)

- Wilson Bilkovich (`lambda {...}.should_change(obj, :message)`)
- Lachie Cox (Initial Rails support and diffing enhancements for Objects and custom formats)
- Lindsay Evans (Patch for RSpec on Rails on Windows)
- Jake Howerton (Port of Kevin Clark's ARTS to RSpec)
- KAKUTANI Shintaro (Patch for coloured progress, Rails patches)
- Rich Kilmer (Underscore syntactic sugar)
- Riley Lynch (Patches for the rake task)
- Pat Maddox (Patch for speccing of routes in Rails)
- Luke Redpath (TextMate bundle)
- Nick Sieger (RSpec on Rails patches for stats)
- Daniel Siemssen (RSpec on Rails patches for rails_spec_server)
- Kouhei Sotu (Emacs patch)
- Nikolai Weibull (Core patches for spec names)
- Mike Williams (Core patches for arbitrary comparisons like `5.should_be < 6`)
- Chad Wooley (Patch to mocks to make it possible to specify raised exception instance)

If you think your name ought to be here, please contact the RSpec team.

Webdesign by JM

私とRSpec

- 
- Team
- Wilson Bilkovich (`lambda {...}.should_change(obj, :message)`)
 - Lachie Cox (Initial Rails support and diffing enhancements for Objects and custom formats)
 - Lindsay Evans (Patch for RSpec on Rails on Windows)
 - Jake Howerton (Port of Kevin Clark's ARTS to RSpec)
 - **KAKUTANI Shintaro (Patch for coloured progress, Rails patches)**
 - Rich Kilmer (Underscore syntactic sugar)
 - Riley Lynch (Patches for the rake task)
 - Pat Maddox (Patch for speccing of routes in Rails)
 - Luke Redpath (TextMate bundle)

• Luke Redpath (TextMate bundle)

• Pat Maddox (Patch for speccing of routes in Rails)

RSpec

- **BDD**実践のためのツール
- Rubyによる**言語内DSL**
 - Domain Specific Language
 - ドメイン特化言語
- **統合テスト**環境

BDD実践のためのツール

- システムの特定の部分が、
- ある文脈(**context**)において
- 期待(**expectation**)される
- 振る舞い(**behaviour**)を
- 仕様化(**specify**)する

Rubyによる言語内DSL

- BDDの**語彙**を活用
 - context, specify, expectation
- プログラミング言語の**構造**に囚われすぎない
 - 1ファイル: nコンテキスト
 - **no TestCase, no test_method**
- **Testing DSL** (角谷ローカル)

統合テスト環境

- **Green Bar** (specコマンド)
- **人に優しい出力** (Testdox-like repots)
- **built-in mocking/stubbing**
- Rakeタスク (自動化)
- RCov (CO カバレッジ)
- diff出力
- RSpec on Rails (流行は大事だ!)

```
context “5ドルの場合” do
  setup do
    @five_dollar = Money.dollar(5)
  end

  specify “2を掛けると10ドルであること” do
    (@five_dollar * 2).should_eq! Money.dollar(10)
  end

  specify “5とは等価ではないこと” do
    @five_dollar.should_not_eq! 5
  end

  ...
end

...
```

意識の変容

さらに興味深いのは、<ヘプタポッドB>はわたしのものの考えかたを変えていくという事実だった。

テッド・チャン 『あなたの人生の物語』

ノイズが

少ない

設計に集中

できる

そのまま

書ける喜び

**GreenBar原理主義:
Fake Itの多用**

BDDで変わらないもの

馬車重動

駆動

- **設計**による決定と、
動作する**コード**からの
具体的な**フィードバック**
- 開発者の**感情**
 - 不安だ。何かがおかしい。これでいい。

ま と め

BDDとは:

- 顧客と同じ**語彙**を使って記述したシステムの**振る舞い**からの**フィードバック**と、
- 開発者の**感情**によって、
- **駆動**される開発

母語をASCIIで
表現できる人達
は良いですね

日本語が母語の

我々は

どうしよう？

まとめ

- BDDはTDDのマインドセットを**より明確にする**
- BDDはマインドセットを明確にする**語彙とツール**を提供する
- BDDもTDDも、**具体的なフィードバック**と開発者の感情が開発を**駆動**する
- 母語をASCIIで表現できる人は良いですね

「でも、コードはそうするように言っている
みたいですよ。やってみましょう。」

『ケント・ベックのSmalltalkベストプラクティス・パターン』

ご清聴

ありがとうございます

ございました

**Happy
Testing!**

何かご質問は？

Do you have any questions?