Rubyによる
たのしいユーザー基盤再構築

RubyWorld Conference 2017
2017-11-01

諸橋恭介 @moro

Kyosuke MOROHASHI

moro

moro

cookpad


高橋征義＋松田明＋諸橋恭介――著
Masayoshi Takahashi, Akira Matsuda, Kyosuke Morohashi
Rails3
レシピブック
190の技
ActiveRecord
ActionMailer
ActionView
ActionController
Bundler
DRY
定番から旬の一品まで
おいしいテクニックを
たっぷり集めました。
CRUD
RSpec
Rake
ActiveSupport
ActiveResource
ActiveModel
Rack
Ruby on Rails 3.1/3.0対応


はじめる！
Cucumber
諸橋恭介
日本語で書かれた
受け入れテストを
Rubyで実行でき
るようにする。

2017-04-06

編集

ユーザー基盤を作り直しながらRailsでのサービス層に向き合う

こんにちは。パートナーアライアンス部の諸橋 (@moro) です。

突然ですが、わたしはいまクックパッドの「ユーザー基盤」を再構築しようとしています。一口に「ユーザー基盤の再構築」といっても、そのゴールが何を指すかは(わたし自身にとってもまだ)漠然としており、固定されたゴールは見いだせていません。しかし後述するように、いくつかの問題は明確な形を取っています。言い換えると、それら明確な問題と向き合いながら『柔軟でいい感じのユーザー基盤を目指す』というのがこの再構築プロジェクトの目的です。

その第一歩目として、ユーザー登録部分を現状のクックパッド本体とは別の小さなRailsアプリケーションとして実装を進め、つい先日、一部の限定された利用者の方に向けて公開することができました。 今後も様子を見ながら公開範囲を拡大していく予定です。

Rubyによる
たのしいユーザー基盤再構築

Rubyによる
たのしいユーザー基盤再構築

「たのしい」

- ❌ おもしろおかしい
- ⭕ いきいきと前向きに取り組める

“

And there's business value in fun - after all motivation is a major factor in programmer productivity.

それから、「楽しさ」にもビジネス価値があります——結局、モチベーションこそがプログラマの生産性を左右するのです。

https://martinfowler.com/bliki/DynamicTyping.html
(http://bliki-ja.github.io/DynamicTyping/)

“

われわれは機械ではないので、やりたいと思うときに、何かを達成できるんです。何かを達成するには、突き動かす内なる力が必要です。それはモチベーションによって現れるんですね。(略) モチベーションを笑ってはいけない。

「なんでRubyなんか作った!? 迷惑だ！」に対するMatzの答え

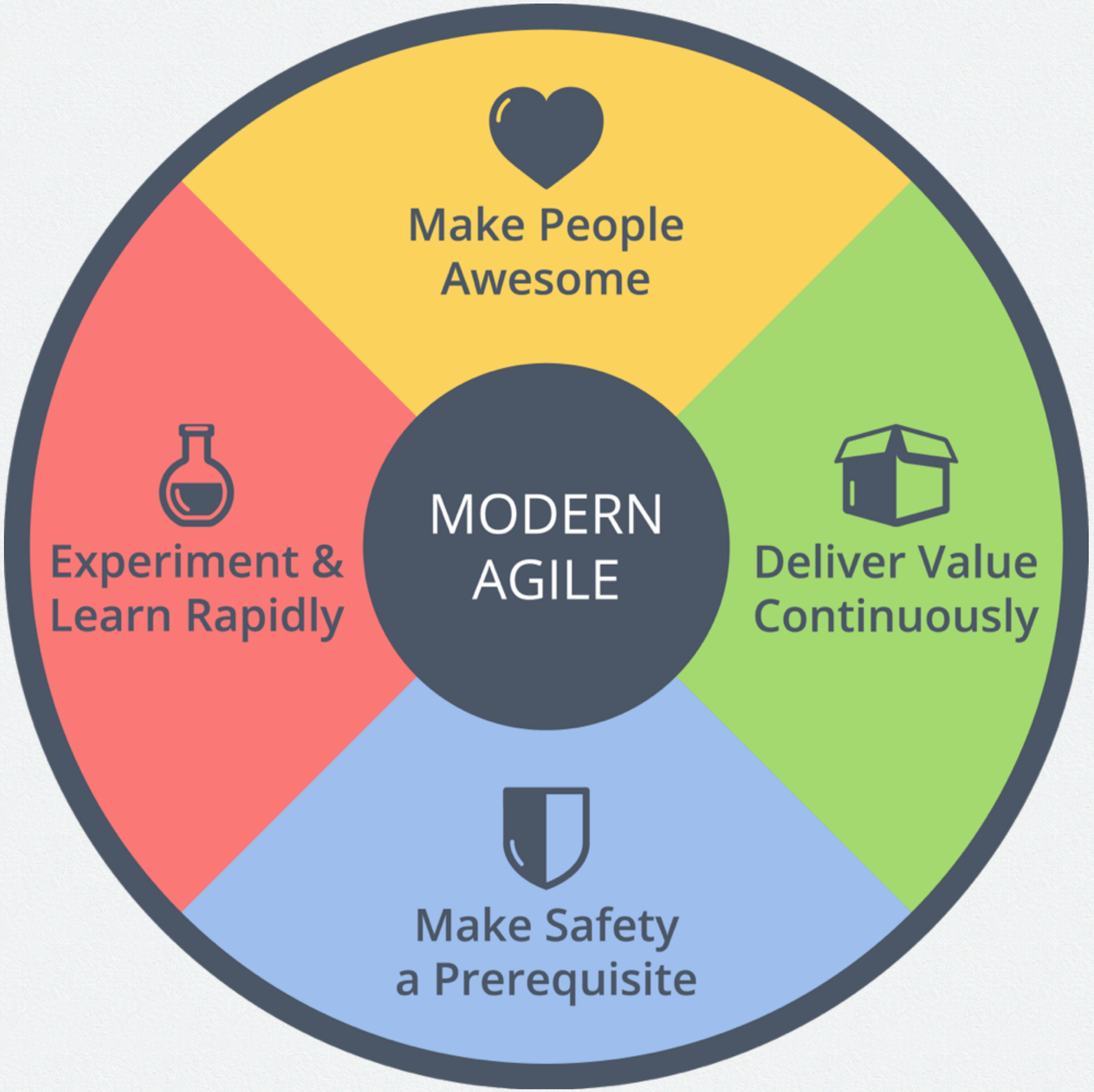
http://el.jibun.atmarkit.co.jp/rails/2012/10/ruby-matz-7080.html

“

つまり「こんなことを言ったらチームメイトから馬鹿にされないだろうか」、あるいは「リーダーから叱られないだろうか」といった不安を、チームのメンバーから払拭する。心理学の専門用語では「心理的安全性（psychological safety）」と呼ばれる安らかな雰囲気をチーム内に育めるかどうかが、成功の鍵なのだという。

グーグルが突きとめた！社員の「生産性」を高める唯一の方法はこうだ
--プロジェクト・アリストテレスの全貌

http://gendai.ismedia.jp/articles/-/48137



- 人々を最高に輝かせる
- 高速に実験&学習する
- 継続的に価値を届ける
- 安全を必須条件にする

Rubyによる
たのしいユーザー基盤再構築

Rubyによる
たのしいユーザー基盤再構築

チームとチームの外の人と
みんなでサービスを届けるのがたのしい

チームで開発するのがたのしい

コードを書くのがたのしい

チームとチームの外の人と
みんなでサービスを届けるのがたのしい

チームで開発するのがたのしい

コードを書くのがたのしい

2017-04-06

編集

ユーザー基盤を作り直しながらRailsでのサービス層に向き合う

こんにちは。パートナーアライアンス部の諸橋 (@moro) です。

突然ですが、わたしはいまクックパッドの「ユーザー基盤」を再構築しようとしています。一口に「ユーザー基盤の再構築」といっても、そのゴールが何を指すかは(わたし自身にとってもまだ)漠然としており、固定されたゴールは見いだせていません。しかし後述するように、いくつかの問題は明確な形を取っています。言い換えると、それら明確な問題と向き合いながら『柔軟でいい感じのユーザー基盤を目指す』というのがこの再構築プロジェクトの目的です。

その第一歩目として、ユーザー登録部分を現状のクックパッド本体とは別の小さなRailsアプリケーションとして実装を進め、つい先日、一部の限定された利用者の方に向けて公開することができました。 今後も様子を見ながら公開範囲を拡大していく予定です。

簡単な問題はあまり残っていない

- 10年前
  - `rails generate scaffold users email:string password_digest:string`
- 現在
  - これまで使ってくれているユーザーさんに迷惑をかけないよう注意をはらいつつ、ビジネスニーズでうまれた多種多様な概念やデータ構造を整理して、今後のサービス展開の基盤となる。

簡単な問題はあまり残っていない

- 月次利用者数：約6000万人
- プレミアム会員数：190万人以上
  - この方たちに迷惑をかけないことを大前提とし、概念を整理して、実装もシンプルにしていく。

※数字は2017年6月末時点

The Recipe for the World's Largest Rails Monolith

Akira Matsuda

https://speakerdeck.com/a_matsuda/the-recipe-for-the-worlds-largest-rails-monolith

意識してたのしくしたい

Ruby

A PROGRAMMER'S BEST FRIEND

Google™ Custom Search | Search

Downloads | Documentation | Libraries | Community | News | Security | About Ruby

Ruby is...

A dynamic, open source programming language with a focus on simplicity and productivity. It has an elegant syntax that is natural to read and easy to write.

Download Ruby or Read More...

```ruby
# The Greeter class
class Greeter
  def initialize(name)
    @name = name.capitalize
  end

  def salute
    puts "Hello #{@name}!"
  end
end

# Create a new object
g = Greeter.new("world")

# Output "Hello World!"
g.salute
```

たのしい Ruby

高橋征義＋後藤裕蔵——著
Masayoshi Takahashi, Yuuzou Gotou
まつもとゆきひろ——監修
Yukihiro Matsumoto

第5版

“

Rubyは「たのしさ」を第一の目標にした世界初のプログラミング言語です。

-- Rubyアプリケーションプログラミング 「1.3 Rubyの哲学」

Rails 10年史

- 昔は新進気鋭のWebフレームワークだった。
- 広く使われた結果、いまは愛憎ともに集める。

最小限の機能から少しずつ進める

再構築を決意したとは言え、サービスにおいて「ユーザー」というのはとても大事かつ複雑なドメインモデルです。

実際に、現クックパッドのActiveRecordモデルクラス(以下ARモデル) `User` はファイル中で定義されるメソッドが328個、 `include` しているモジュールが14個、 `has_(one|many)` 合わせて175個の関連が定義されている3,000行を超えるクラスです。このすべてを一気に新しいユーザー基盤(以下、新基盤)に置き換えようとしても、影響範囲が大きすぎてリリースにたどりつけないのではないかという懸念がありました。

そこで今回は最小限の機能をリリースすることを重視し、従来より負担の少ないユーザー登録フローを実現するアプリケーションを開発しました。

現方式では(1-1)ファーストビューで必要な情報をすべて入力したあとで、(1-2)到達確認のためにメールを送信し、(1-3)メール中のURLにアクセスすると登録が完了するという流れで登録します。対して新基盤では、(2-1)ファーストビューではクライアントサイドのみの処理とし、(2-2)

レガシーコード?

- 文字通りの「レガシー」= 遺産
- 私たちのビジネスは、このコードがお金を稼いでくれているからこそできている。

“

Regardless of what we discover, we understand and truly believe that everyone did the best job they could, given what they knew at the time, their skills and abilities, the resources available, and the situation at hand.

この会では、プロジェクトの全員が置かれた状況下でベストを尽くした、という ことを疑ってはならない (平鍋訳)

http://www.retrospectives.com/pages/retroPrimeDirective.html

Rails 10年史

- Ruby で昼の仕事ができるようになった。
- それ自体が、私たちにとってはとても偉大な遺産。
- ちゃんと、アップデートをしていく。

“

‘Use all the wise and knowledge you have of your own, Sam’

「お前が自分で持っている智恵や知識をある限り使ってごらんよ、サム、」

-- 指輪物語「王の帰還」

Is Rails dead?

- Fat Controller, Fat Model …
- Validatezer や Hanami が流行っている

この10年の学び:

- ツールやフレームワークをただ適用するだけではうまくいかない。
- 背景になっていそうなコンセプトを、自分たちでちゃんと試してみよう。

サービス層としてのForm objectをよく考えてみる

```
 9  11    end
10  12
    13  +  def activate!
```

moro 3 days ago **Owner**

従来の設計だと、これはform objectに切り出していた種類の処理かなと思いますが、こっちに寄せた意図はありますか?
(字義通りの (HTML) form じゃないけど)

shota-iguchi 3 days ago • edited **Member**

そうですねー、ここがformオブジェクトかどうかがよくわからなくなってきていて、formオブジェクトは何かしらの入力値のバリデーションと次の処理の実行を行うものだと考えていて、ここはバリデーションを行う必要がないと思うのでformオブジェクトというよりはサービスクラスなのかな・・・みたいな感じで悩んでいました。

ひとまずは処理がそんなに膨らまない想定だったのでここに書いてみました。

moro 3 days ago **Owner**

> formオブジェクトというよりはサービスクラスなのかな・・・みたいな感じで悩んでいました。

うーん、なので「(字義通りの (HTML) form じゃないけど)」サービスクラスとしても使っているわけなので、ARモデルからは剥がしませんか? というのがコメントの意図でした。

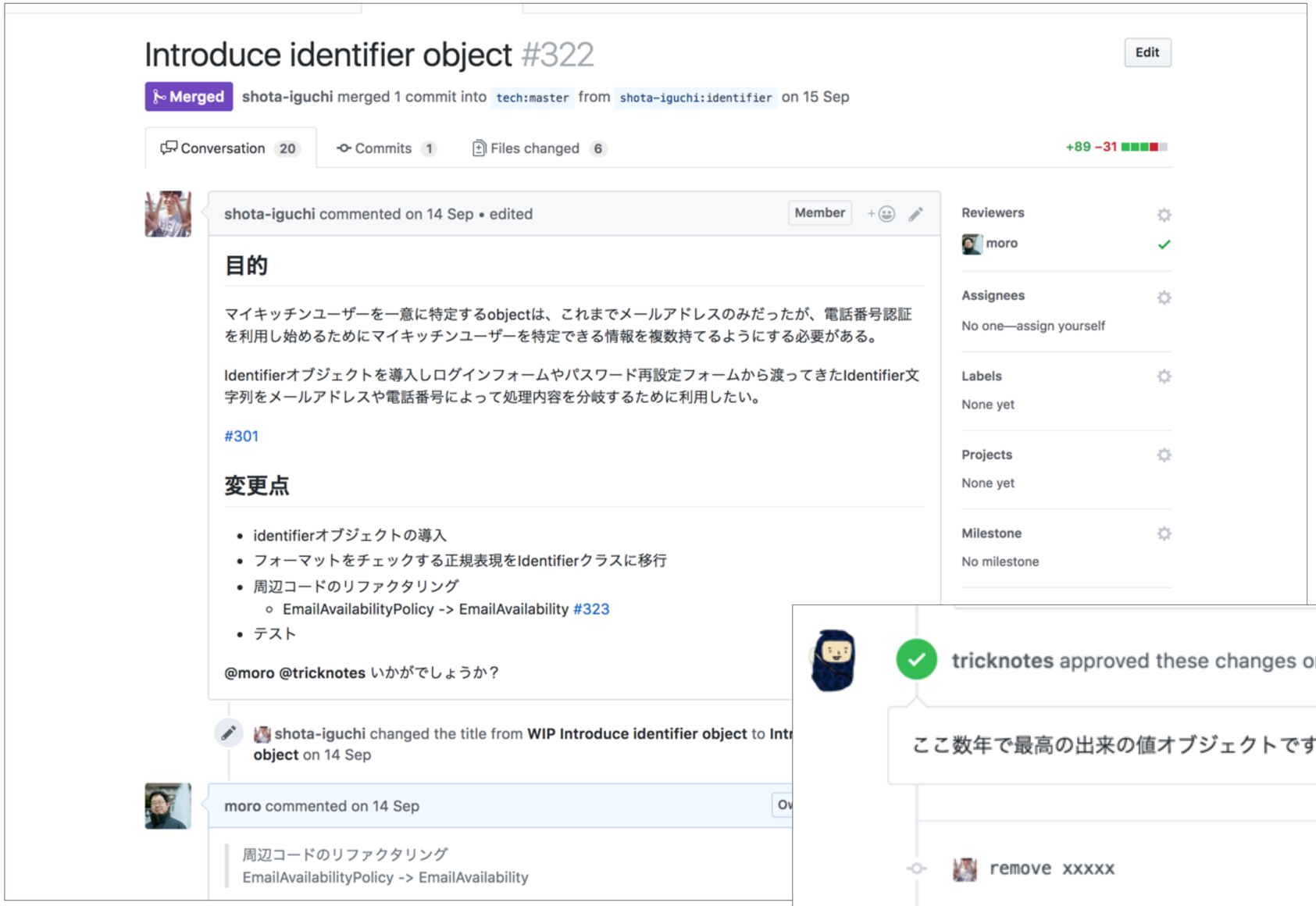
> ひとまずは処理がそんなに膨らまない想定だったのでここに書いてみました。

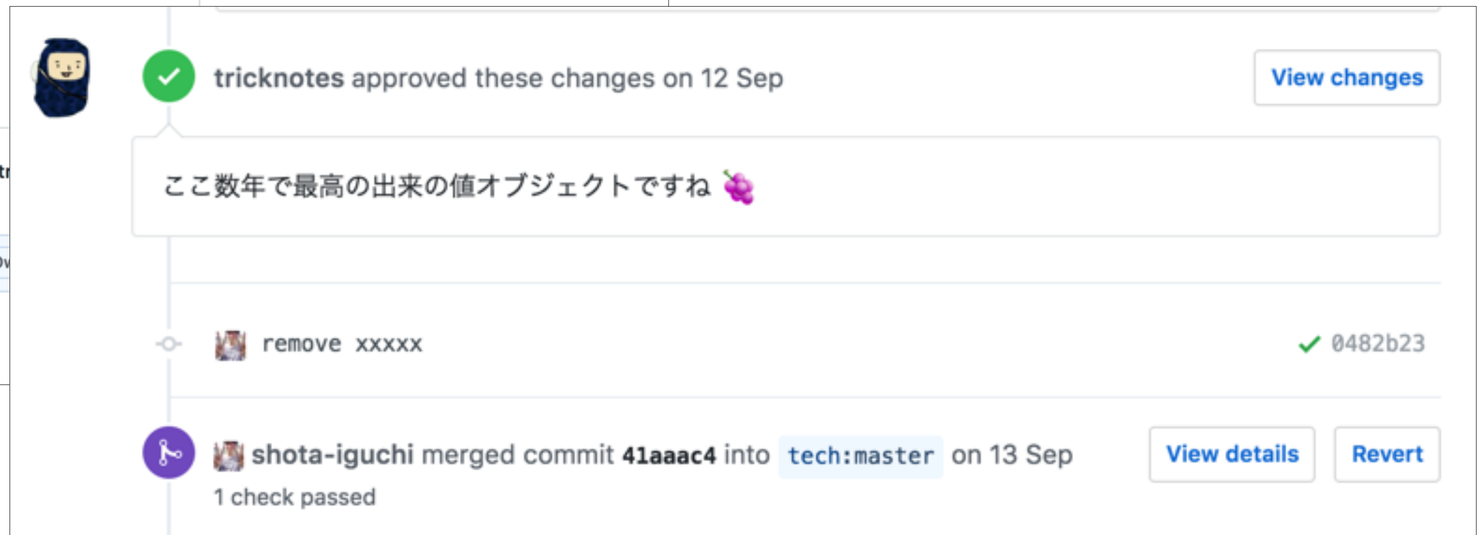
念のため、行数それ自体というよりも、複数テーブルへの書き込みはここのARクラスから分離しておきたいなあという気持ちです。

サービス層としての Form Object

- テスト可能な「モデル層」は有用。
  - 「プレゼンテーションとドメインの分離」ってやつ
- コントローラからは常に **who** と **when** が渡される。
  - `ActiveSupport::CurrentAttributes` なるほど。
- だんだん「フォーム」じゃないことも分かってきた。
  - サービスを「入れ子」にしたいこともある。

Value Objectを発見する





これまでのビジネスからも学ぶ

- 登録中のデータは「`validate: false` なユーザー」ではない。
- 登録それ自体も一つの大事な機能である。
  - `Registration` という概念を発見した。
  - 同時に、登録後にちゃんと元のコンテキストに戻すのもとても大事ということも、学んだ。

```
resources :email_registrations, only: %i(show new create) do
  collection do
    get :thanks
  end

  resource :user, controller: :cookpad_users,
    only: %i(show new create), module: :email_registrations
  end
end
```

- Eメールでのユーザー登録 (email_registrations) 自体をリソースとして扱う。
- そこから別リソースとしてユーザー (cookpad_users) を作る。

コードを書くのは
たのしい

チームとチームの外の人と
みんなでサービスを届けるのがたのしい

コードを書くのがたのしい

“

‘Use all the wise and knowledge you have of your own, Sam’

「お前が自分で持っている智恵や知識をある限り使ってごらんよ、サム、」

-- 指輪物語「王の帰還」

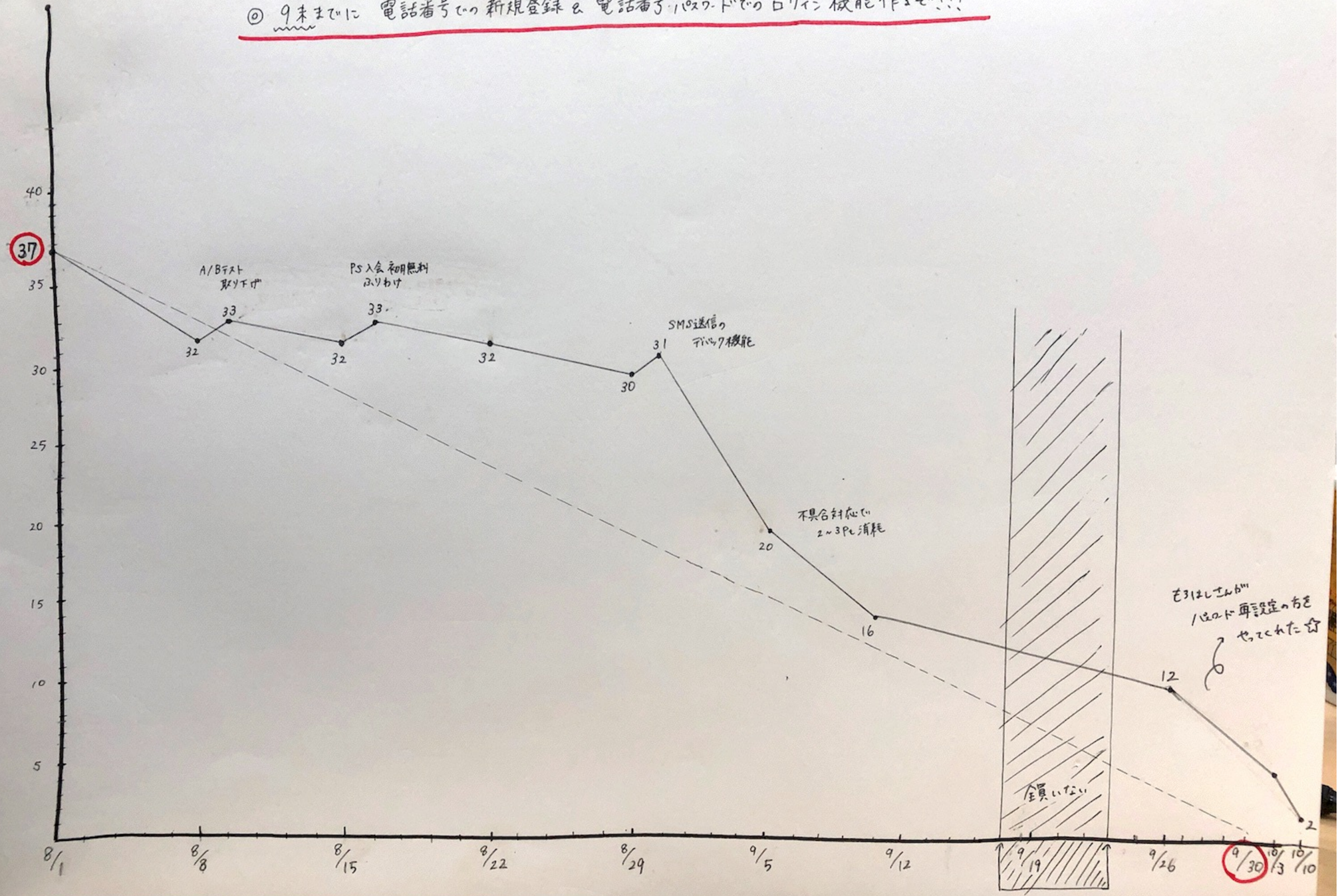
いろんなプラクティスがあるけど、ちょっとずつ試した:

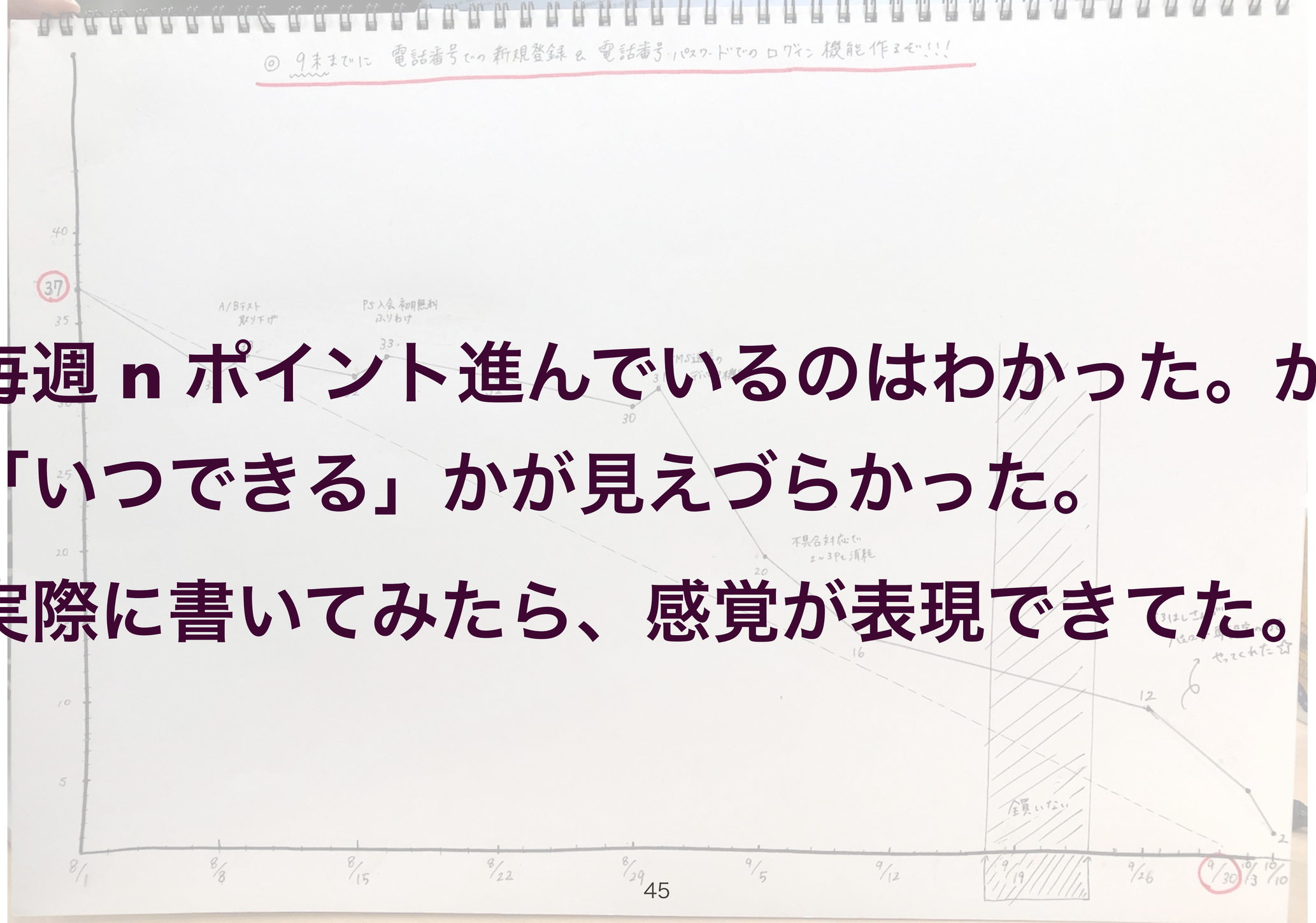
- ふりかえり
- 見積りと計画づくり

いろんなプラクティスがあるけど、ちょっとずつ試した:

- ふりかえり
  - これまでに何をしたか
- 見積りと計画づくり
  - これから何をするか

◎ 9末までに 電話番号での新規登録 & 電話番号・パスワードでのログイン機能作るぞ!!!





- 毎週 n ポイント進んでいるのはわかった。が、「いつできる」かが見えづらかった。
- 実際に書いてみたら、感覚が表現できてた。

携帯番号登録をみんなで触る会・第二回をうけてのフィードバック #342

Edit New issue

Open tricknotes opened this issue 16 days ago · 0 comments

tricknotes commented 16 days ago • edited Member

10:03 16:30~ の触る会で出てきたフィードバックをとりあえず issue にしました。
前回の issue はこちら #315

- ログイン画面 -> #311 (comment) で対応予定
  - [ ] 連続サブミットできないようにローディングを出す
  - [ ] 送信中はボタンを disabled にする
  - [ ] エラーが返ってきた場合に良い感じのハンドルを行う
  - [ ] 「パスワードを忘れた方はこちら」が足りない
- 電話番号登録の認証画面
  - [x] 認証コードを規定回数間違った場合に「"クックパッドサポートにお問い合わせください" は表示されては」が表示されてはいけない (キャプチャ) #344
- パスワードリセット画面(`password_resets/show`)
  - [x] パスワード入力欄がバリデーションエラーの際に赤く囲われない #346
  - [x] placeholder のフォントがおかしい(placeholder には monospace いらない) #343
- パスワードリセットの認証画面(`password_resets/new`)
  - [x] placeholder のフォントがおかしい(placeholder には monospace いらない) #343
  - [x] 「SMSが届かない場合」がない #349
- AuthCenter
  - [ ] 携帯番号でユーザー登録しようとすると、「username parameter is required.」というエラーになる (Sentry)

❤️ 2

This was referenced 15 days ago

Use default font-family for placeholder #343 Merged

Fix condition for error message that needs support #344 Merged

Assignees
No one—assign yourself

Labels
None yet

Projects
None yet

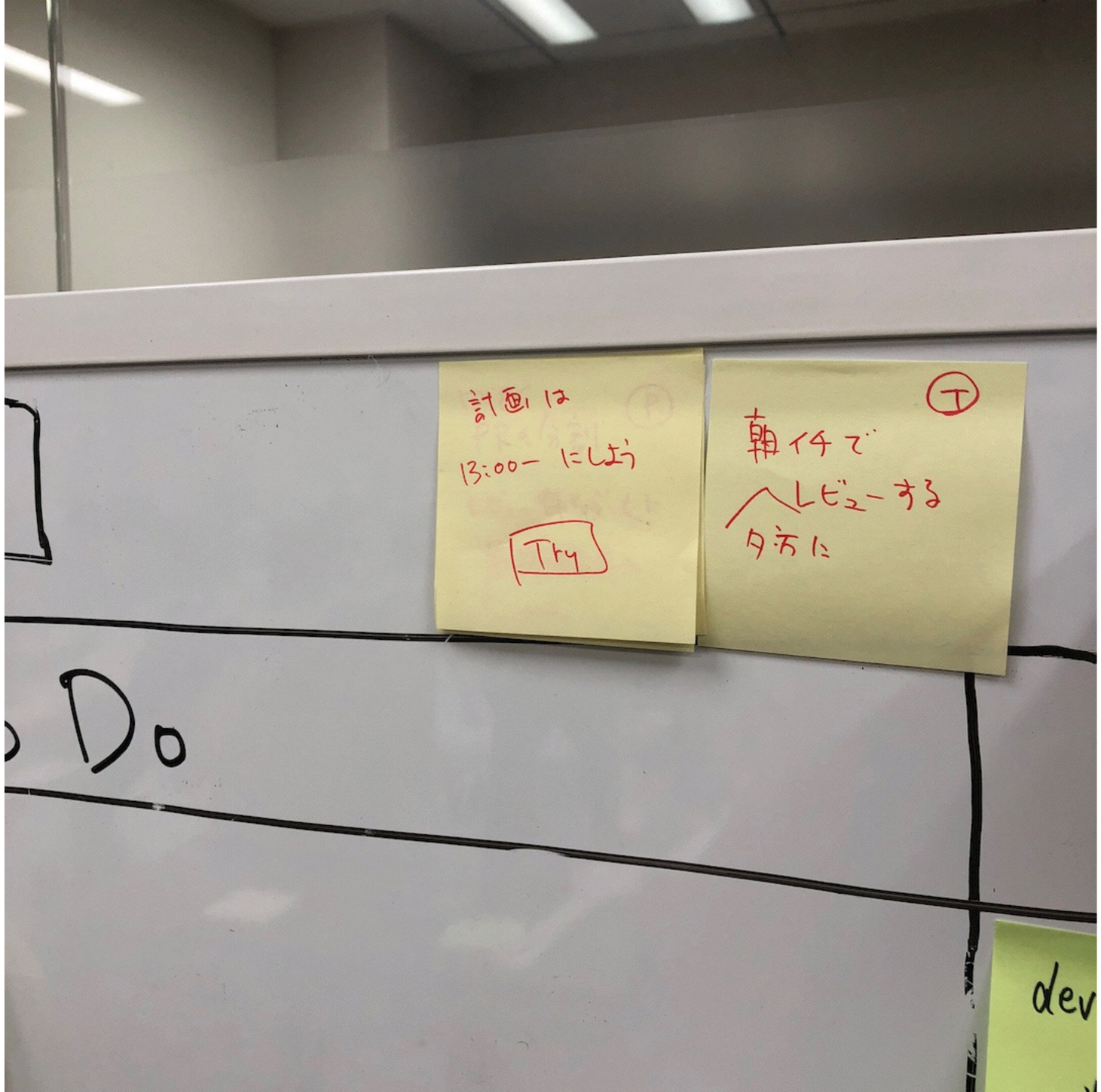
Milestone
No milestone

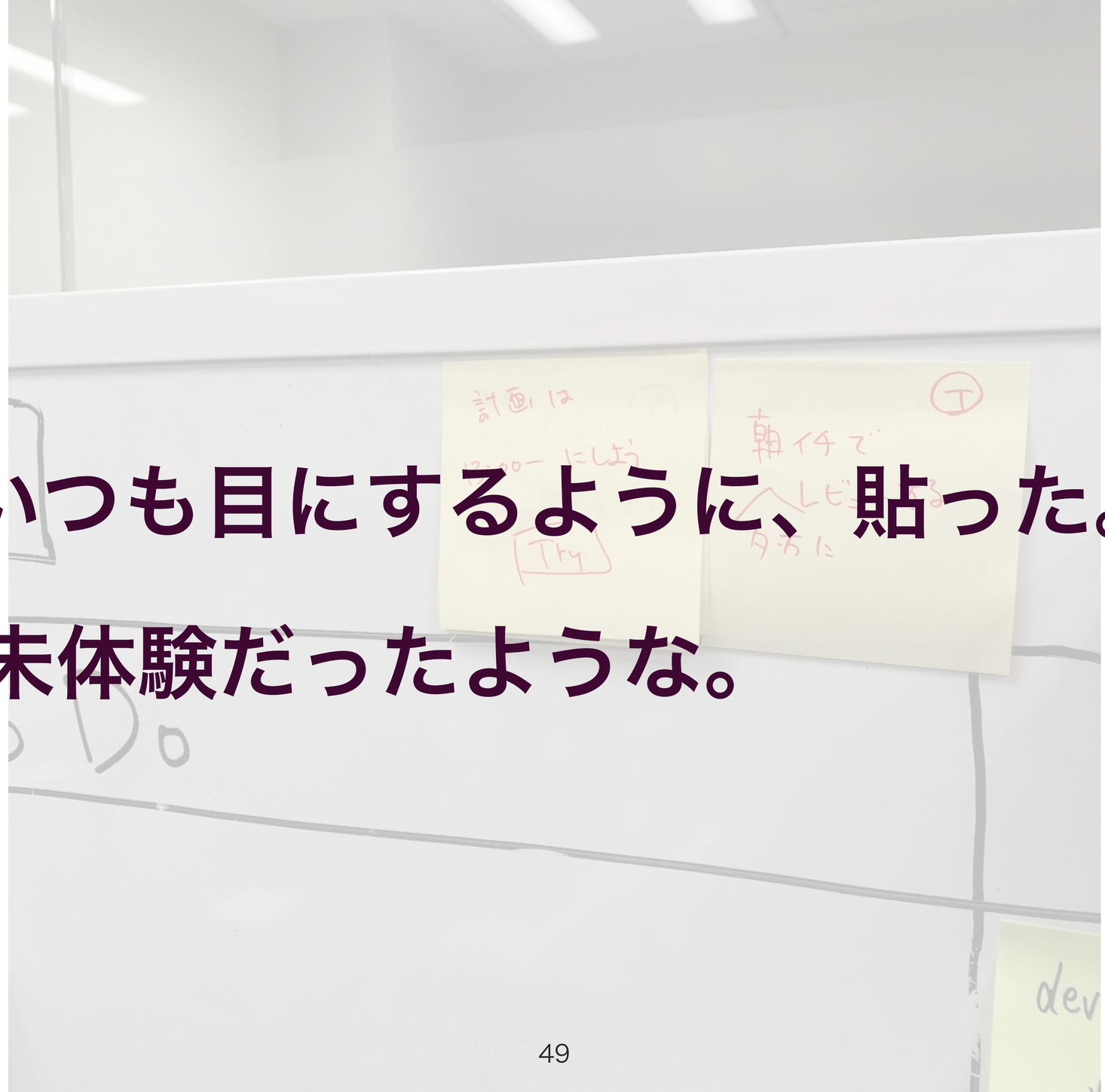
Notifications
Unsubscribe
You're receiving notifications because you're subscribed to this repository.

1 participant

Lock conversation

- 社内のプラクティスを輸入した。
- ドメインエキスパートにさわってもらう。
- 大小様々の気づいたことをどんどん直す。


計画は
13:00ー にしよう
Try
(工)
朝イチで
レビューする
夕方に
Do
dev



- Tryをいつも目にするように、貼った。
- これは未体験だったような。

チームで開発するのがたのしい

コードを書くのがたのしい

似ている

- 「これまで」に学んで
  - Rails一般やアプリケーションコードの苦労
  - 機能の使われ方
- 「これから」を考える
  - ちょっとずつ「よい」設計をする
  - 必要なら技術的負債を借り入れる

似ている

- 「これまで」に学んで
  - これまでのイテレーションでのチームの進め方。
  - 作ろうとした機能でほんとうに必要だったもの。
- 「これから」を考える
  - **Try**
  - これまでを踏まえた見積りと計画づくり。

自己相似性がある

チームとチームの外の人と
みんなでサービスを届けるのがたのしい

チームで開発するのがたのしい

コードを書くのがたのしい

「知っていた」こと

“

だが、高みに 到達することが目的であれば、ソフトウェア開発は「プログラマーとその他大勢」 で成立するものではない。それが、過去 5 年間で私が学んだことだ。

— エクストリーム・プログラミング 「第25章 結論」

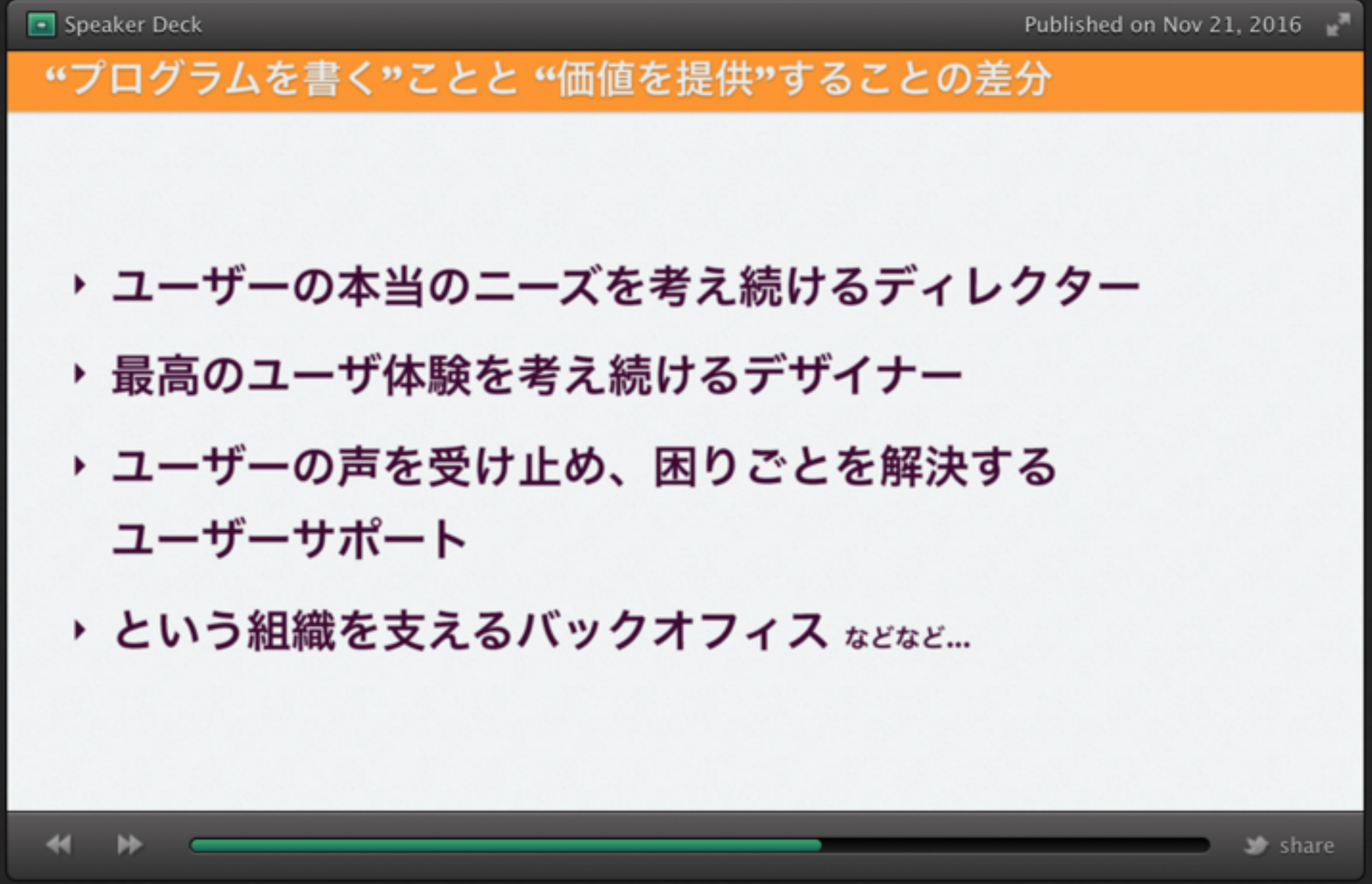
“

There's a difference between

knowing the path and walking the path.

道を知ることと歩きだすことは違う

-- Morpheus; Matrix



達人プログラマーとわたし/PragProg and Me

by MOROHASHI Kyosuke

Published November 21, 2016 in Programming

“

XPの価値は、ビジネスの世界で実践すべきものだ。快適に暮らせる生活を送るだけでなく、関係者全員がお互いにリスペクトした人間関係によって、新しい働き方を開発するのである。(略) 創造的にいきいきと働くのである。

— エクストリーム・プログラミング 「第25章 結論」

自分たちのビジネスのユーザー状態を学習できてきた

- かつては正しかった前提 (メールアドレスがある、など) が必要以上に広範に適用されている。
  - かつては「メールアドレスを入力してユーザー登録」を前提としていた。
  - メールアドレス入力を省いて、有料会員として使いたいと言ってくれるユーザーさんも出てきた。
  - 「アプリをインストールしてくれた」らユーザーと扱う仕組みがいるのでは。

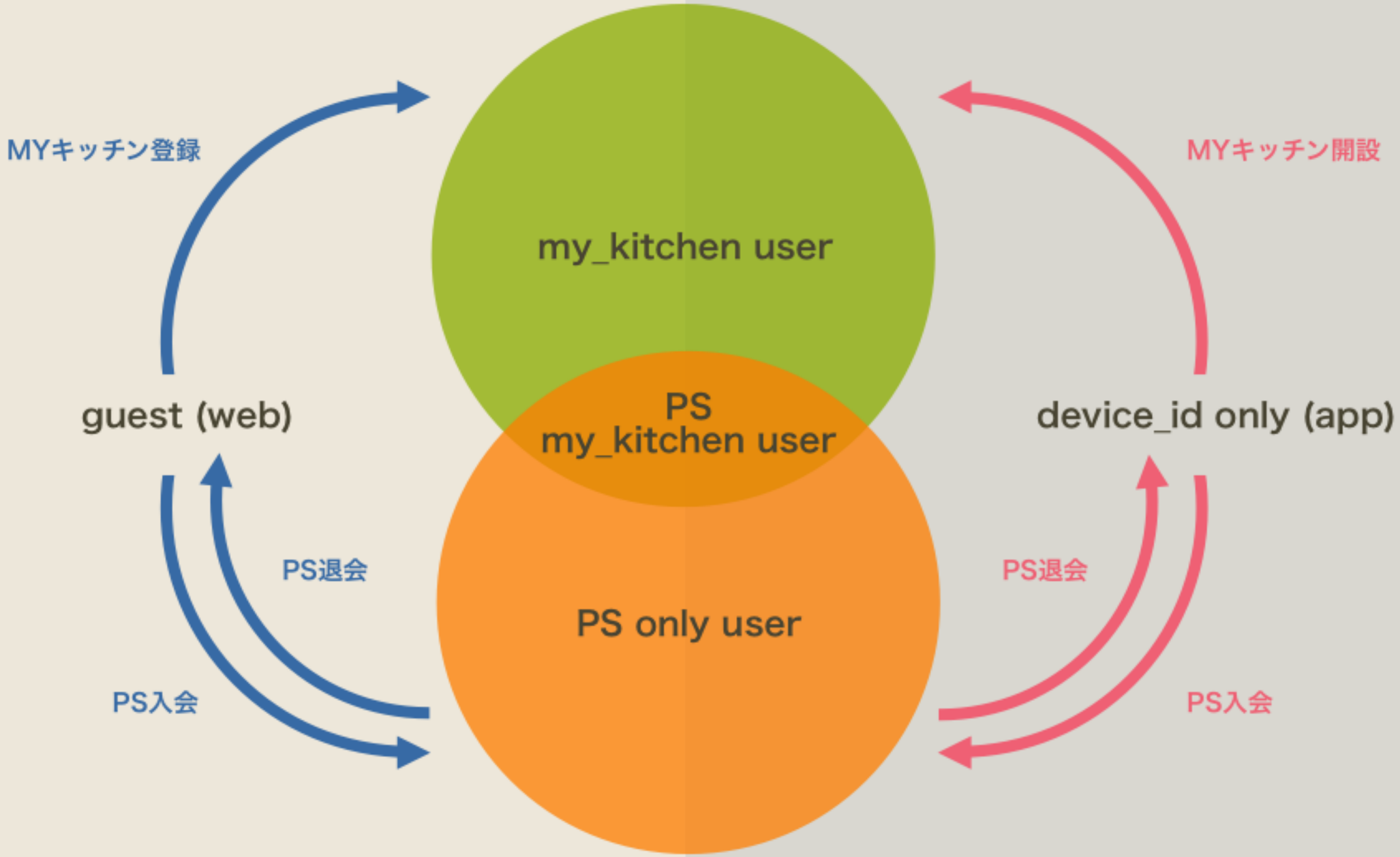
データ状態が難しい

- サービス開発の各所で扱いづらくなっている。
  - コミュニケーション系の機能で決済手段を意識しないといけない、など。
  - 企画やサポートも難しければ、実装上の落とし穴になることもある。
  - 新しく入社した人が理解するのも難しい。

“

チームが意識的に努力すれば、ドメインモデルはその共通言語の基盤となれるし、その一方で、チームのコミュニケーションをソフトウェアの実装と結びつけることもできる。

ドメイン駆動設計 「第2章 コミュニケーションと言語の使い方」


MYキッチン登録
my_kitchen user
MYキッチン開設
guest (web)
PS
my_kitchen user
device_id only (app)
PS退会
PS only user
PS退会
PS入会
PS入会

“

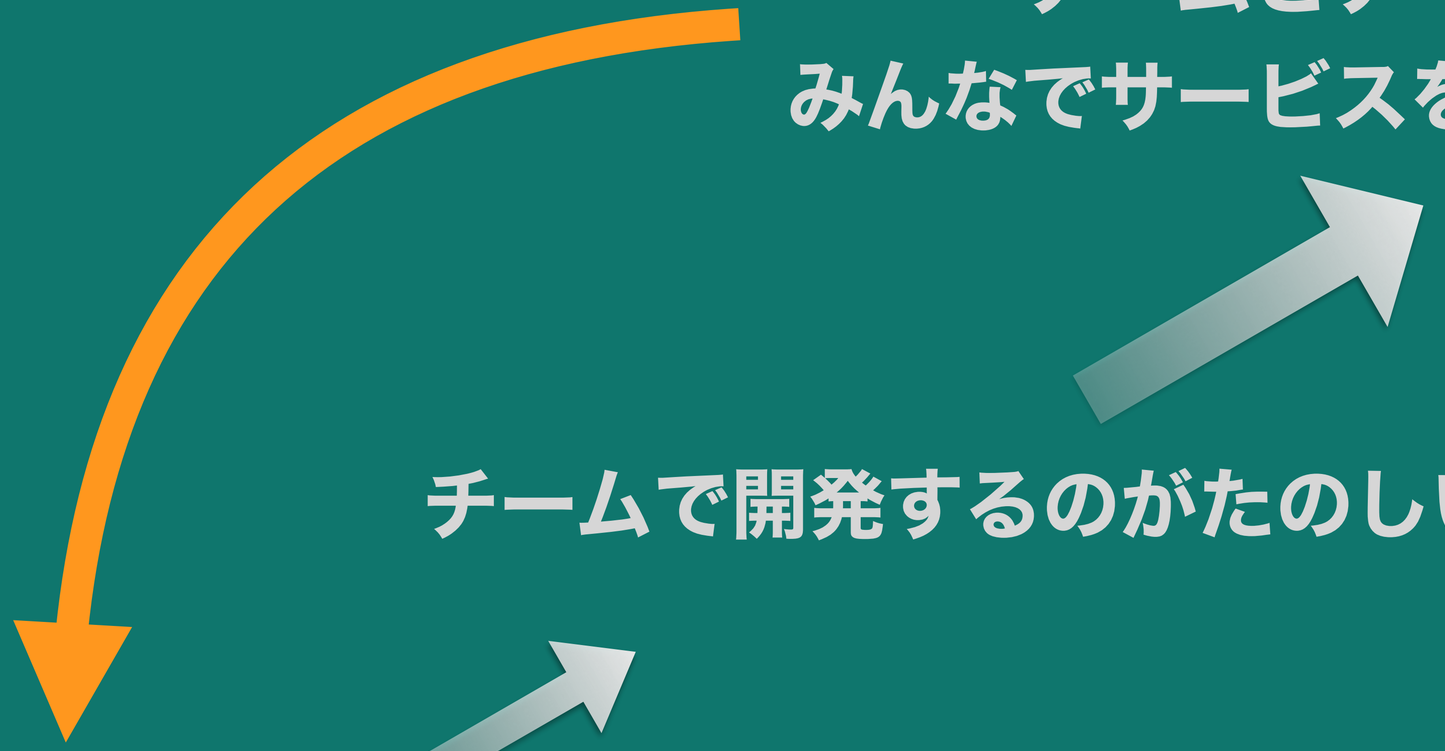
そのプロセスのなかで、開発者として最善を尽くすことだ。ビジネスのためになる優れたコードを書くことだ。

-- エクストリーム・プログラミング 「第1章 XPとは何か」

チームとチームの外の人と
みんなでサービスを届けるのがたのしい

チームで開発するのがたのしい

コードを書くのがたのしい



チームとチームの外の人と
みんなでサービスを届けるのがたのしい

チームで開発するのがたのしい

コードを書くのがたのしい

最小限の機能から少しずつ進める

再構築を決意したとは言え、サービスにおいて「ユーザー」というのはとても大事かつ複雑なドメインモデルです。

実際に、現クックパッドのActiveRecordモデルクラス(以下ARモデル) `User` はファイル中で定義されるメソッドが328個、 `include` しているモジュールが14個、 `has_(one|many)` 合わせて175個の関連が定義されている3,000行を超えるクラスです。このすべてを一気に新しいユーザー基盤(以下、新基盤)に置き換えようとしても、影響範囲が大きすぎてリリースにたどりつけないのではないかという懸念がありました。

そこで今回は最小限の機能をリリースすることを重視し、従来より負担の少ないユーザー登録フローを実現するアプリケーションを開発しました。

現方式では(1-1)ファーストビューで必要な情報をすべて入力したあとで、(1-2)到達確認のためにメールを送信し、(1-3)メール中のURLにアクセスすると登録が完了するという流れで登録します。対して新基盤では、(2-1)ファーストビューではクライアントサイドのみの処理とし、(2-2)

簡単な問題はあまり残っていない

- 月次利用者数：約6000万人
- プレミアム会員数：190万人以上
  - この方たちに迷惑をかけないことを大前提とし、概念を整理して、実装もシンプルにしていく。

※数字は2017年6月末時点

負担が減るであろうと見込んでいます。

また、新基盤のデータの持ち方やインフラ構造についても、現行クックパッドからの漸次的な改善を進めていくために、以下のような方針を取りました。

- 新基盤では、現クックパッドとDBを共有する。
  - 新しいフローでのみ必要なデータは新規のDBに、現クックパッドで使うデータは現DBに書き出す。
  - 現DBに書く場合でも、ビジネスロジックは新たに書き直す。
- 今後もインクリメンタルな移行を進めるため、現サービス https://cookpad.com/ 以下の特定のパスをリバースプロキシで分岐して新基盤と統合する。
- 技術要素は社内推奨かつ新しいものを採用する。
  - 現クックパッドからの移行がしやすく、開発者の熟練度も高いRuby on Railsを採用する。
  - 技術的な進歩にキャッチアップするため、Ruby 2.4 + Rails 5.1.rc1 + Webpack + React などで開発する。
  - ECS上にアプリケーションをデプロイするためのHakoなど、社内の標準的なインフラ構成に乗せる。

このように「最小限の機能で、ちゃんと動くソフトウェア」をリリースすることを優先した結果、ユーザー登録のみを行うアプリケーションとして最初のリリースを迎えられました。

アプリケーションを少しずつ設計する

次の「レガシー」をつくっているのかも...

- 技術的負債を借り入れながら進める
  - 一時的にDBの同テーブルへ複数アプリケーションから writeするとか
- 今回もその中でベストを尽くす
  - 機能のスコープを分けてちょっとずつ出す。
  - 変数のスコープも分ける、テスト可能にする... etc

“

There's a difference between

knowing the path and walking the path.

道を知ることと歩きだすことは違う

-- Morpheus; Matrix

意識してたのしくしたい

“

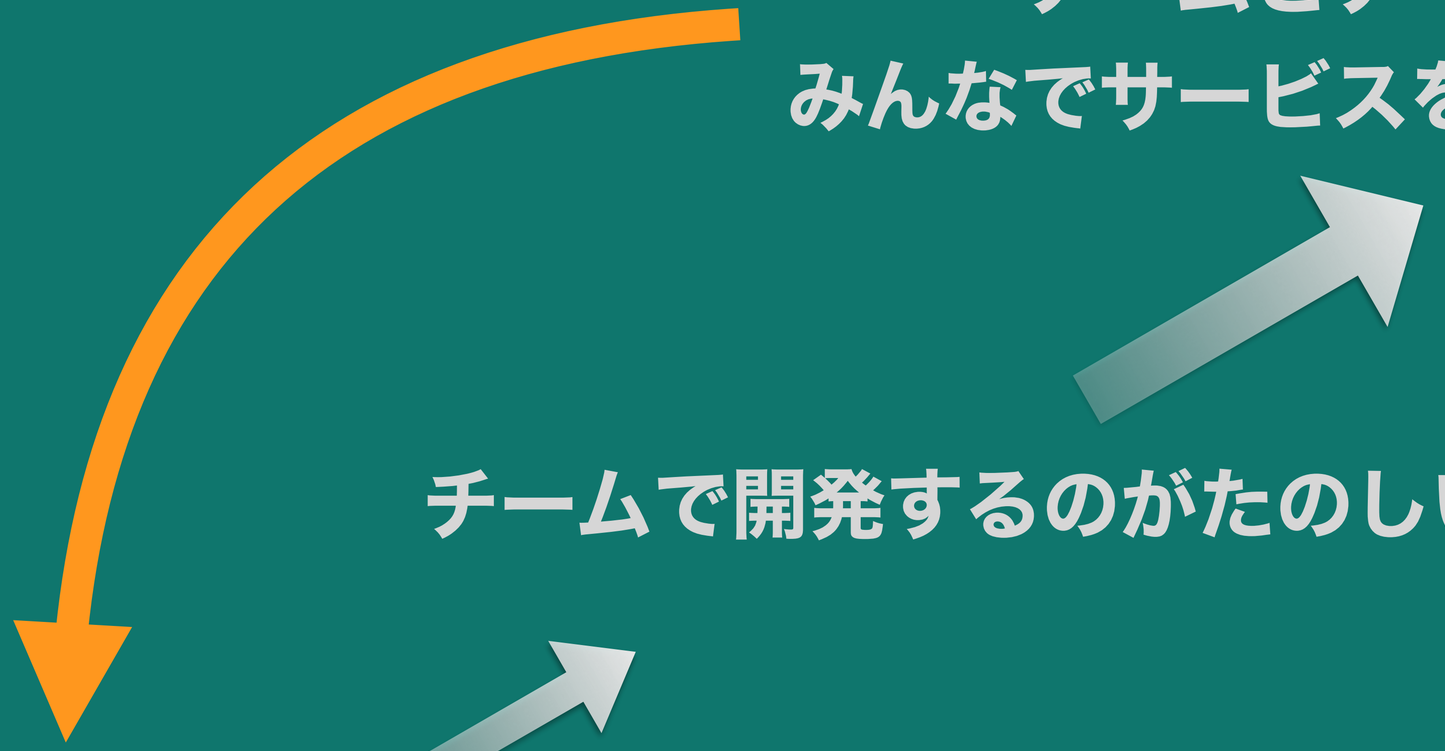
それから、「楽しさ」にも
ビジネス価値があります

http://bliki-ja.github.io/DynamicTyping/

たのしい Ruby

高橋征義＋後藤裕蔵——著
Masayoshi Takahashi, Yuuzou Gotou
まつもとゆきひろ——監修
Yukihiro Matsumoto

第5版



チームとチームの外の人と
みんなでサービスを届けるのがたのしい

チームで開発するのがたのしい

コードを書くのがたのしい