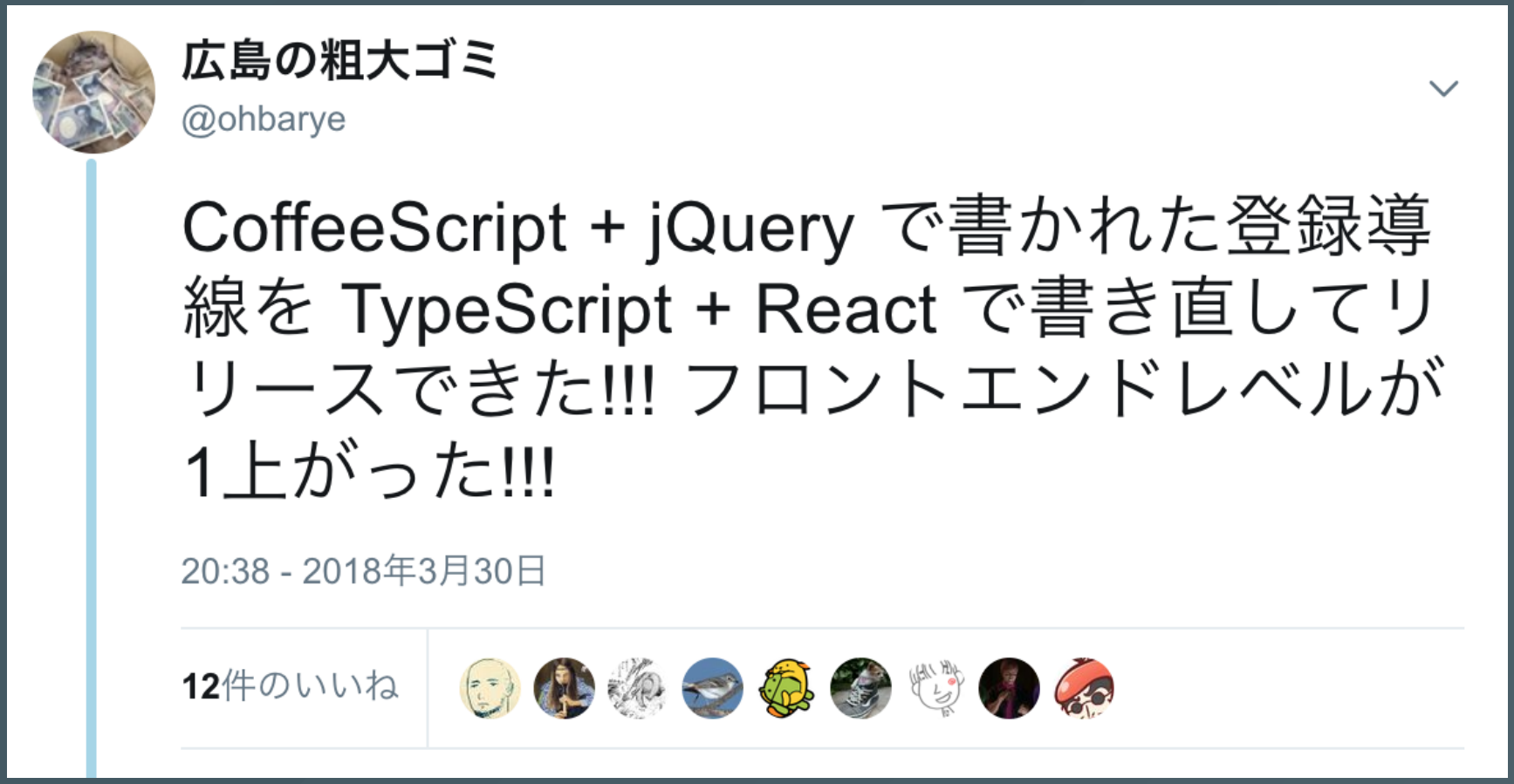
フロントエンドのレベル上げ

Rails エンジニアが Webpacker を使う場合

by @ohbarye at Meguro.rb#15

本日の内容について


広島の粗大ゴミ
@ohbarye
CoffeeScript + jQuery で書かれた登録導線を TypeScript + React で書き直してリリースできた!!! フロントエンドレベルが1上がった!!!
20:38 - 2018年3月30日
12件のいいね

Content

0. Webpacker とは？
1. なぜレベルアップする必要があったのか
2. Railsフロントエンドのモダン化
3. Webpacker pros/cons

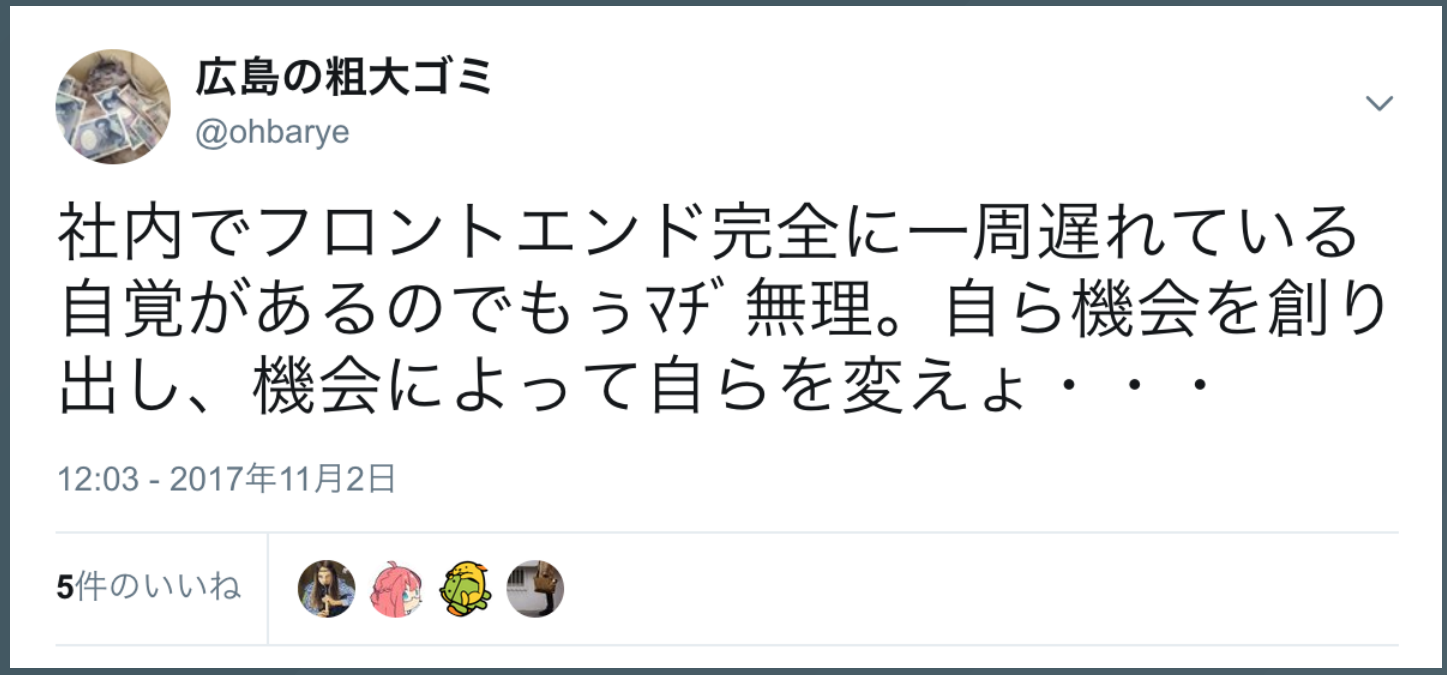
0. Webpacker とは？

Webpacker おさらい

- Webpacker は Webpack と Rails を統合する gem
- コマンド数発でいきなり ES6, TypeScript, React で Rails のフロントエンド開発を始められる
- config ファイルも開発用と本番用を生成
- Rails organization で開発されている
- => 本当に良いツールか?　というのは後半で振り返る

1. なぜレベルアップする必要があったのか

"front-end complex"

- React, TypeScript が社内で標準になる流れ
- React Native 人材になっていく同僚たち
- 

ビジネスとして

- 当時担当の web アプリは決済サイト (＋登録導線)
  - 決済は堅牢であることが最も大事
  - 一方、このサイト内の登録導線は CVR に直結
- フロントエンドは Rails おなじみのスタック
  - jQuery + CoffeeScript
- 拡張が苦しく、複雑な仕様やA/Bテストに対応するには作り直したほうが早い

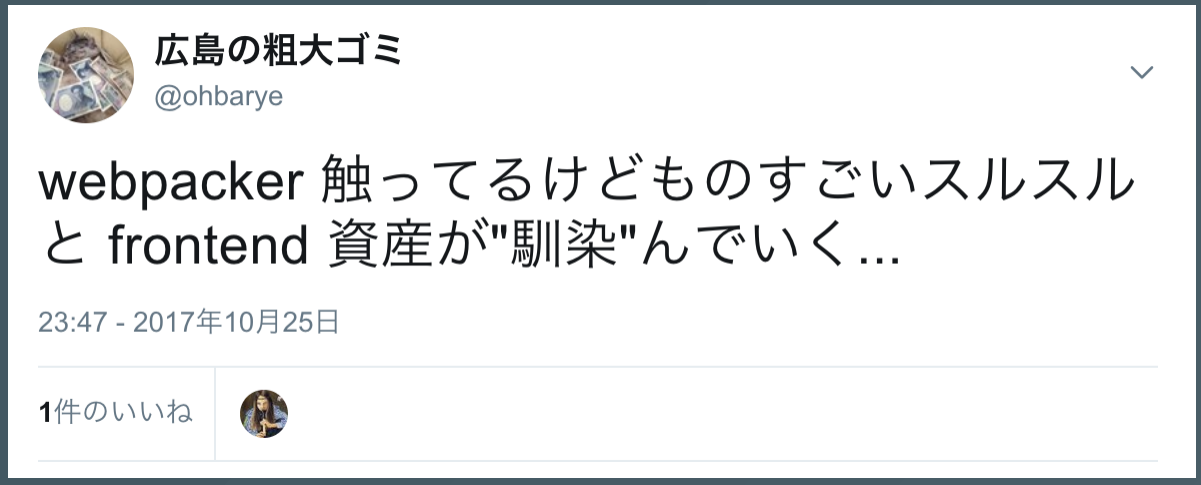
2. Railsフロントエンドの
モダン化

step1: Introduce Webpacker

- 手始めに Webpacker を install
  - CircleCI で使う docker image を更新
  - 同僚 @mtsmfm が `docker compose up` で webpack-dev-server も立ち上がるようにしてくれた
- Heroku Buildpack を使ったデプロイで躓いたが、バックエンドの知識でなんとかなった

step2: Code migration

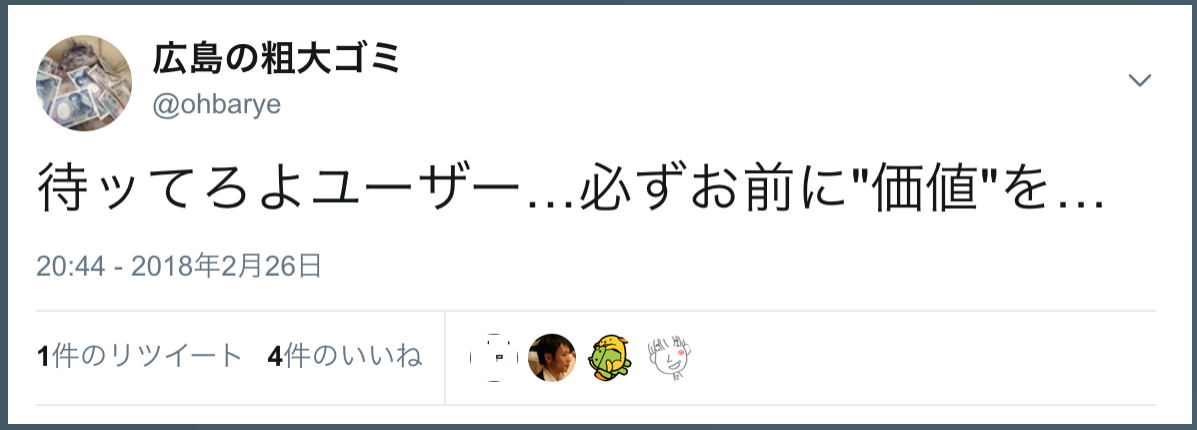
- Webpacker 管理下にファイルを移動 (rm manifest)
  - app/assets/javascripts => app/javascripts
- asset pipeline gem を npm ライブラリに置換
- CoffeeScript を TypeScript に書き換え
-



step2: Code migration

- できる人ならこうした設定なども一瞬で終わるかもしれないが、それでは自分の学びにならない
- 一方、足回りを独力でちまちま進めているといつまでもマージされない、リリースされない
  - リリースされないコードに価値はない...
- 緩やかな移行を実現できたので、**スコープを絞りながら学ぶことが出来た**

step3: SPA

- Partial single page application パターンで実装
  - 登録導線の数画面だけを SPA として実装
  - AWS のマネジメントコンソールも P-SPA
- 単なる書き直しでなくCVR向上のための機能追加を約束
- 

step3: SPA

- 登録画面の数画面のためだけに Redux は不要
- formik
  - 揮発性の高い form データを扱う
  - Dan Abramov 先生が2017/12に言及
- yup
  - validation rule を宣言的に定義
  - JSON schema 的

3. Webpacker pros/cons

Pros

- ハッシュ付きファイル名の自動生成
  - Sprockets と同じ運用ができる
- `rake assets:precompile` にフックしてビルド (`webpacker:compile`) が行われる
  - Sprockets を使っていない場合は `assets:precompile` の alias になる
  - 既存のビルドシステムやアセット管理が活きる

Cons

- Webpacker 固有の知識が求められる
  - webpacker.yml, watched_paths etc.
  - Ruby 経由で webpack を実行する問題
- webpack v4 が出ても Webpacker が対応するまでアップデートできない
  => 既視感... asset pipeline と同じ話では...？
- 余計なレイヤーが1つ増えたとも言える

Webpacker is momentum builder

- "simple" ではなく "easy" を体現するツール
- Simple is not easy

> 「Easy」は勢いを生み出すものです。「Easy」によって早期に得られる低リスクの成功体験は継続へとつながるでしょう。

Webpacker Philosophy

- 公式レポジトリに届いたお便りへの回答
- 開発者の幸福にフォーカスしている

> But the most important benefit that we often overlook is our happiness and experience as programmers. If things are integrated and part of one workflow, then it leads to better and happy work. I guess that's what been the mantra of Rails stack - to provide integrated systems that work together.

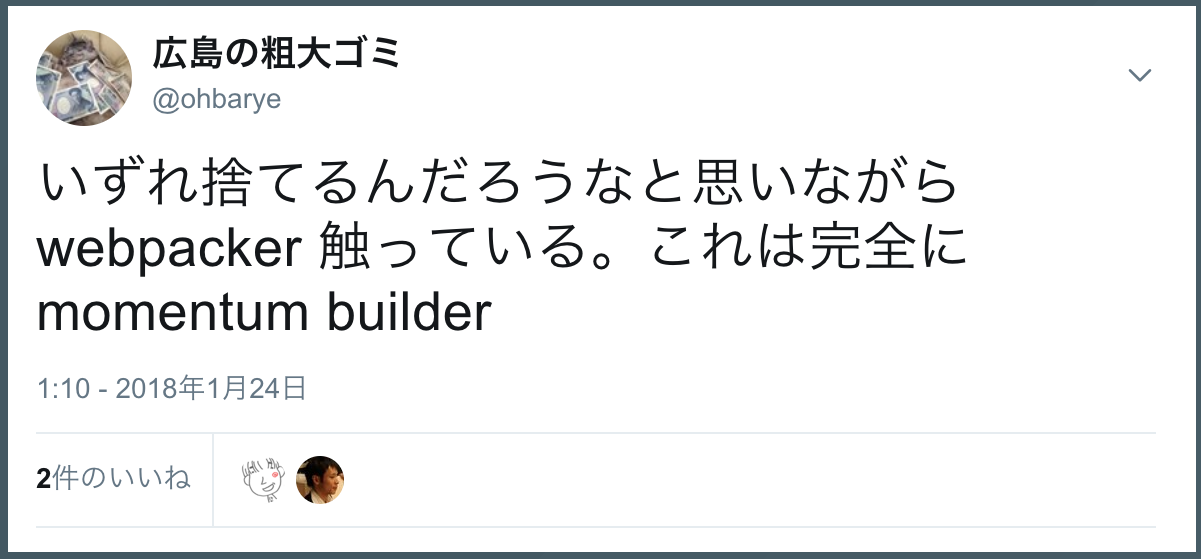
When to use, when NOT to use

- Q. 結局 Webpacker って必要？
- A. 個人的な見解は...
  - 既存プロジェクトのゆるやかな移行には
  - フロントエンドに詳しいエンジニアなら不要
  - 今なら個人開発でも使わない...かな

Webpacker 関連 良記事

- Introducing Webpacker https://medium.com/statuscode/introducing-webpacker-7136d66cddfb
- How we switched from Sprockets to Webpack https://rossta.net/blog/from-sprockets-to-webpack.html
- 【保存版】Rails 5 Webpacker公式ドキュメントの歩き方+追加情報 https://techracho.bpsinc.jp/hachi8833/2018_05_17/56568

今後

- 卒業宣言「Webpacker は補助輪」
- 

- `webpacker:eject` 的なコマンドがほしい
- sass も webpack で管理したい

まとめ

- Webpackerを用いたRailsフロントエンドモダン化を通じてレベルアップできた
- レベルアップと同時にビジネスにも貢献
  - CVR 16.0->18.6% (局所的には 36.5->43.7%)
- 開発を通じて周辺エコシステムに興味・関心++
  - GitHub trending で JS, TS もウォッチするように
  - Webpacker, formik, node, yarn, DefinitelyTyped などの JS エコシステムに contribute できた

自己紹介

@ohbarye : Web Developer / Engineering Manager

Working for Quipper (and we're hiring!　)

http://ohbarye.me/

完