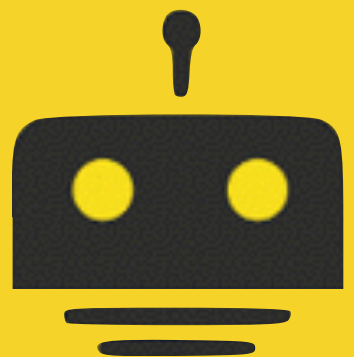




# ラピットプロトタイプینگ

#NodeBots\_jp vol3  
dotstudio @n0bisuke

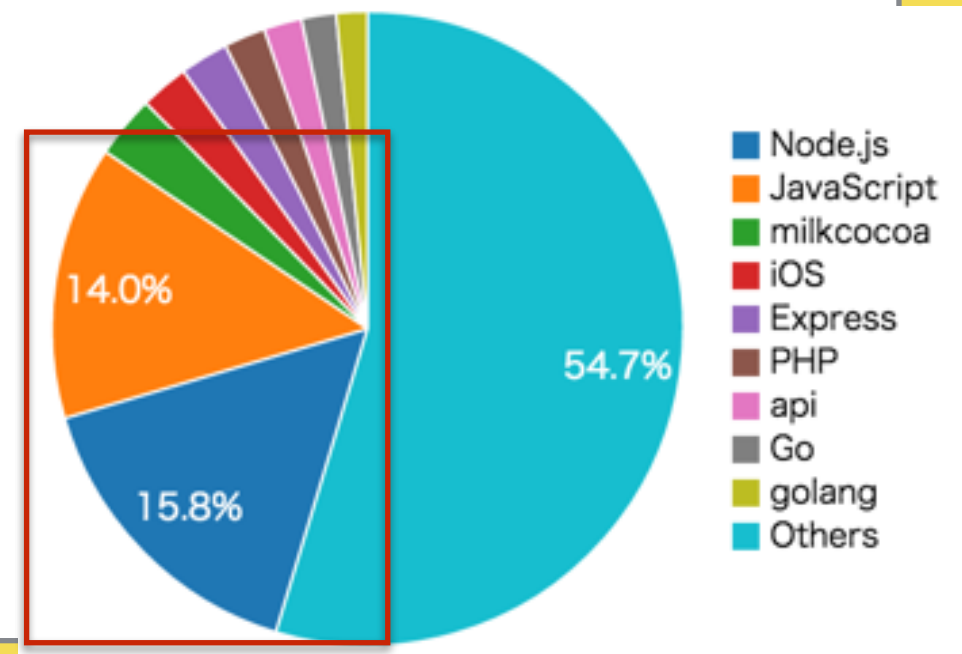


# メモ

- 自己紹介など 10分
- Webを作る時に利用する技術や最近のトレンド 15分
  - フロントエンド 5min
  - バックエンド 5min
  - 運用周り 5min
- ラピットプロトタイピング 40分
  - ビジネスサイドとクリエイティブサイドのジレンマ
  - MVPを考える
  - 実際に僕らが試行錯誤していること
  - 使えるツール
- まとめ 5分
  - まとめる

# About Me

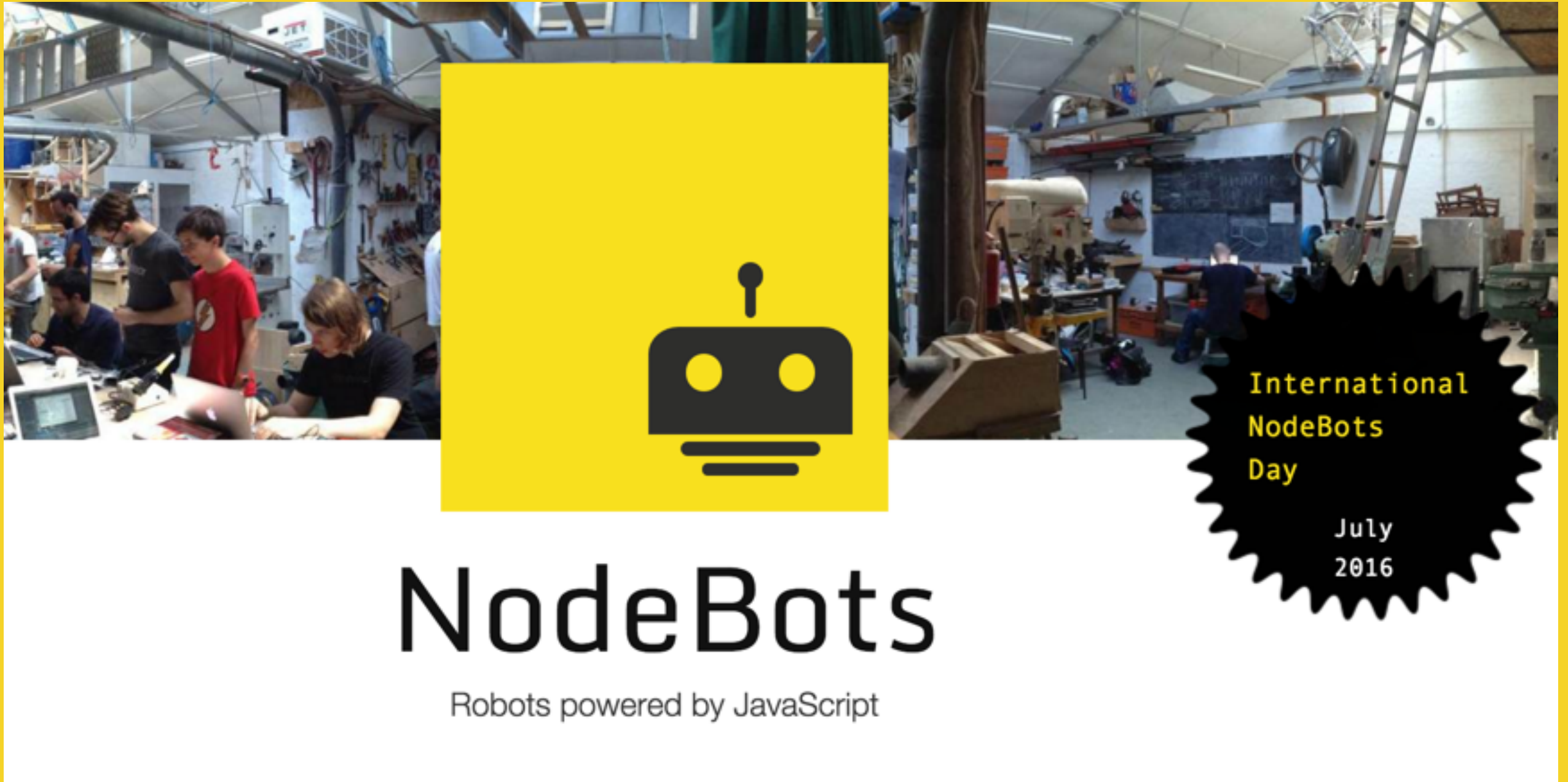
- @n0bisuke
- dotstudio株式会社 代表取締役
- 最近の興味: IoT / JavaScript Robotics
- 宮城県栗原市
- 岩手県立大学 学部/修士
- Milkcocoaエバンジェリスト
- #IoT LT #nodebots



# IPU時代 ダイジェスト

- ・ 2008年入学 g031g089
  - ・ IS研 -> 現佐々木研
- ・ ~ 2009年頃 大学祭実行委員会
- ・ 2010年頃 ソフトウェア情報学部学生会
- ・ 2010~12年頃
  - ・ 高木先生の弟子に
  - ・ ご当地検定プロジェクト
  - ・ 学会とかめっちゃ行ってた
- ・ 2012~2014年頃 g231k020 (うろ覚え)
  - ・ 大学院
  - ・ 台湾留学
  - ・ 学長賞

# NodeBots



NodeBots

Robots powered by JavaScript

International  
NodeBots  
Day

July  
2016

**<http://nodebots.io>**

# **1. Webを作る時に使用する技術や最近のトレンド**

# フロントエンド バックエンド 運用

# フロントエンド



# Webのフロントエンドとは

- ・ HTML, CSS, JavaScript, デザイン, UI設計などの領域
- ・ いわゆる“表側”
- ・ 最近(2012年頃～)生まれた領域のイメージ

# フロントエンド関連職種

- デザイナー
  - WebサイトやサービスのUI/UX設計
  - 必要であればマークアップも
  - フロントエンドエンジニアリングには含まれない
- コーダー / マークアップエンジニア
  - HTML, CSSを駆使してデザインをコードに
  - デザイナーが兼務することも多い
- フロントエンドエンジニア
  - ブラウザ側のJavaScriptが主戦場
  - HTML, CSSなども触る
  - サーバー側との連携も考えるためサーバー側の知識もある程度必要

# フロントエンドエンジニアスキル

- ブラウザ
- レスポンシブ/アダプティブデザイン
- MVCフレームワーク
- タスクランナー
- Node.js

# フロントエンドの知識

- ・ HTMLプリプロセッサ / テンプレートエンジン

- ejs
- ect.js
- Jade

- ・ CSSプリプロセッサ

- sass
- less
- styls
- (postcss)

- ・ CSSデザインパターン

- smacss
- bem

- ・ AltJS

- typescript
- coffeescript
- hexe
- dart

# フロントエンドの知識

- MVCフレームワーク
  - backbone.js
  - Angular.js / Angular2
  - React/Redux
- AltJS
  - typescript
  - es2015 / babel
- タスクランナー・トランスパイラ
  - grunt, gulp
  - webpack, rollup, browsrify
  - bower, npm

# バックエンド

# Webのバックエンドとは

- サーバーサイドのアプリケーションの実装
- サーバー/インフラを整備
  - セキュリティ
  - ドメイン
  - インフラの選定
- 開発運用周り
  - コンテナ
  - テスト自動化
  - CI

# バックエンドスキル

- ・ サーバーサイドのアプリケーション
  - 言語: PHP, Ruby, Python, Node.js, Golang, Scala ...
  - フレームワーク
  - セキュリティまわり
    - 各種攻撃対処 (XSS,
    - SSL対応
  - インフラ
    - AWS, GCP, Azure



# インフラトレンド

この三つは押さえておこう

- **AWS**

- Amazon Web Service
- 世界で一番使われてるインフラ
- EC2, RDS, S3など

- **GCP**

- Google Compute Platform
- Google のサービス
- Google App Engine, BigQuery

- **Azure**

- Microsoftのサービス
- Webapps, Azure Function

# 運用

# ここでのいう運用

- **コンテナ管理、CIなどよりも管理周り**
  - チャットツール
  - ソースコード管理
  - その他必要なツール

# チャットツール

- **Slack**
- **Hip Chat**
- **Chat Work**
- **idobata**
- **Gitter**

# ソースコード管理

**基本はGitです。Gitが使えないとはじまらない**

- **ホスティングサービス**
  - **GitHub**
  - **BitBucket**
- **自分でホスティングするタイプ**
  - **GitLab**
  - **GitBucket**

# その他ツール

- オンラインストレージ
  - DropBox
  - Google Drive
- ドキュメント
  - Googleスプレッドシート
  - MS Word
  - Keynote
- Markdown
- 画像圧縮/加工
  - image optic
  - Photoshop
  - Gyazo
- デザインプロトタイプツール
  - Pratt
  - cache
  - sketch

**一般的にWeb界隈で利用してる  
技術やトレンドの紹介でした。**

# この辺の勉強にオススメメディア

- Web媒体

- CodeZine
- [gihyo.jp](http://gihyo.jp)
- LIG
- HTML5Experts
- CodeGrid
- Qiita

- 紙媒体

- WebDB Press
- Webデザインニング
- Software デザイニング

- ポッドキャスト

- リビルド.fm
- [mosaic.fm](http://mosaic.fm)



## 2. ラビットプロトタイピング

さっきまでは一般的にWeb界隈で利用してる技術やトレンドの**一部**紹介でした。

さっきまでは一般的にWeb界隈で利用してる技術やトレンドの**一部**紹介でした。

けっこうありますよね。

# Web界隈の技術は常に進化していて イケてる技術トレンドがすぐに変わる

- ・ 今使ってる技術トレンドが1年後に生き残ってるかは不明
- ・ 特にフロントエンドがカオス
- ・ この状態下でも技術選定は慎重にやらないと痛い目に
- ・ **プロダクトやチームの状況にあわせてどの技術を選定するかが重要**

# Web界隈の業態分け

## Web界隈

### 制作会社

- ライゾマティクス
- シフトブレイン
- チームラボ
- カヤック
- LIG

### 事業会社

- はてな
- 楽天
- pixiv
- Retty
- ランサーズ

業態によって働き方が異なり、利用ツールも変わっていく

# Gitホスティングサービス選定の例

## 制作会社

- プロダクトが大量に並走
- リポジトリ数が増える
- 案件に関わる開発人数はそこまで多くない



**ユーザー数課金の  
BitBucket**

## 事業会社

- プロダクト数は1つや数個
- リポジトリ数はそこまで膨らまない
- 人数は増えていく



**リポジトリ数課金の  
GitHub**

\*最近はGitHubも料金形態が変わってるみたい

**今日の話のバックグラウンドは  
新しくサービスを立ち上げる時の  
話がメインになります。**

# ビジネスサイドとクリエイティブサイドのジレンマ



- **ビジネスサイド**
  - 経営、マネージャー
  - お金を生み出す、コストを減らす
- **クリエイティブサイド**
  - エンジニア、デザイナー
  - プロダクトを設計し、作る側

## よくある会話

> エンジニア

GitHubとSlackの有料版使いたい

> マネージャー

そこにコスト掛けたら、その分売り上げ上がるの？

## 両方大事

- ・ エンジニア/デザイナーがいなければモノは作れない
- ・ 売る人がいなければ、エンジニア/デザイナーの給料は出せない

# 仕事でプロダクト立ち上げ時に何が必要か

# 仕事でプロダクト立ち上げ時に何が必要か

お金

# 仕事でプロダクト立ち上げ時に何が必要か

- ・ **設計**

- **ビジネス**

**どれくらいの期間で利益を作るか**

**どのマーケットにリーチするか**

- **クリエイティブ**

**運用しやすいアーキテクチャ**

**ユーザビリティ**

# エンジニアは割と全体が見えてない人が多い

- ・ 言語やフレームワーク選定はビジネス全体からみたらだいぶ隠れた部分

- ・ いいものを作れば給料が上がるは幻想

(とはいえアーキテクチャ選定は重要)

# MVPを考えてみよう



- ・ 設計が正しかったかどうかは**検証をしない**  
**と分からない**
- ・ 通常PDCA:  
**仮説** -> マネタイズ/システム設計->システム開発-> ユーザー獲得-> FB -> **検証** -> 改修 or 再設計 -> 再開発
- ・ “設計->開発”ってどれくらい時間かけられる…?

- ・ リーンスタートアップ的な考え方
- ・ 設計と開発は時間がかかるので、ミニマムスタートでサービスリリース
- ・ 仮説 -> 検証までのスピードを重視
- ・ このシステム/機能ならやっていける！という確信を早くもちたい

# 実際に僕らの試行錯誤を紹介します

## 2. ラビットプロトタイピング > 実際にやってること

2016.07.19

### dotstudio株式会社 設立のご報告



のびすけ



<https://dotstud.io>

## 2. ラビットプロトタイピング > 実際にやってること

最新情報   ウェアラブル   コネクテッドカー   スマートホーム   スマートシティ   ドローン   ロボ   工場   店舗・オフィス   要素技術   IoT技術部

© 2016.07.26 06:05

# IoTエンジニアをささえるセレクトショップdotstud.io（β）がオープン

いいね！ 94

ツイート

G+1 0

B! 1

LINEで送る

Pocket 10

BETA


studio

ドットスタジオ

IoTなセレクトショップ

PROJECTS

BLOG



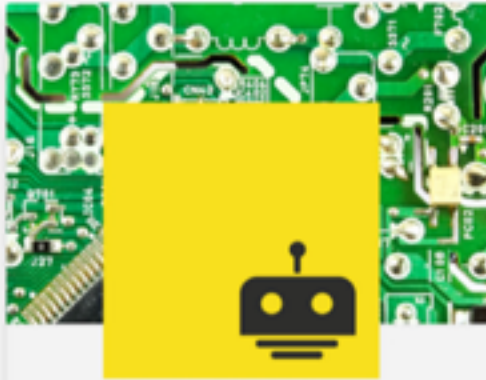
2016.07.01

アクセサリで電子工作デビューしない？  
暗闇で光るフービヤリングを作ってみた



2016.07.01

Wio Nodeを使って自作のソーシャルIoTラジコンを作ろう



NodeBots

2016.07.01

Webな人でもハードウェア制御が簡単  
に！ Node.jsでJavaScript Roboticsを楽し

IoTTLTというIoT縛りの勉強会が昨年から開催されている。17回目を数える同会は最近では200～300名の参加者がある盛況ぶりだ。

# IoTのセレクトショップ

# 開発の前！

## 会社を始める上で考えたこと

- ・ **まずは死なない計画**

人件費とかその他固定費やばい

外部とのアライアンスや営業

- ・ **タイミング/スピード**

今、このタイミングで外に出したいよね



# Webサイトの構築

- **どんな機能が必要と考えたか**
  - **EC機能**
    - **ものを売る**
    - **ものを紹介する**
  - **会社サイトとしての機能**
    - **会社概要や問い合わせ**
    - **法的な記載**
    - **会社の情報発信（メディア）**

# Web構築で考えた内容/アーキテクチャ

- ・ **インフラ選定**

- 日本語の情報が多いか
- “中の人”との関係値
- AWS …?

- ・ **サーバーサイド**

- 枯れてる技術かどうか
- チームメンバーが使える技術
- 処理速度
- Node.js . . . ?



# Web構築で考えた内容/アーキテクチャ

- フロントエンド

- react/redux
- EJS
- SASS
- webpack

- 運用周り

- Dockerでコンテナ運用
- Circle CI で自動デプロイ
- Slack連携

# 実際のスタートでは…

- **インフラ選定**

- 日本語の情報が多いか
- “中の人”との関係値
- AWS …?

- **サーバーサイド**

- 枯れてる技術かどうか
- チームメンバーが使える技術
- 処理速度
- Node.js . . . ?

- **フロントエンド**

- react/redux
- EJS
- SASS
- webpack

- **運用周り**

- Dockerでコンテナ運用
- Circle CI で自動デプロイ
- Slack連携

# 実際のスタートでは…

## ・インフラ選定

- 日本語の情報が多いか
- “素人”との関係
- AWS

## ・サーバ

- 枯れた技術かどうか
- 手頃なマシンで使える技術
- 導入のハードル
- Node.js . . . ?

## ・フロントエンド

- react/redux
- EJS
- SASS
- webpack

## ・運用周り

- Dockerコンテナ運用
- CircleCIで自動デプロイ
- Slack連携

# 実際のスタートでは…

## ・インフラ選定

- 日本語の情報が多いか
- “人”との関係がどうか
- AWSのサポートがどうか

## ・サーバー

- 枯れた技術かどうか
- 最新の技術を使える技術かどうか
- 導入のハードルがどうか
- Node.js . . . ?

## ・フロントエンド

- react/redux
- EJS
- SASS
- webpack

## ・運用周り

- Dockerコンテナ運用
- CircleCIで自動デプロイ
- Slack連携

バックエンド側はがっつり削りました

# バックエンドの必要性を考える

- ・インフラ選定

- 日本語の情報が多いか
- “中の人”との関係値
- AWS …?

- ・サーバーサイド

- 枯れてる技術かどうか
- チームメンバーが使える技術
- 処理速度
- Node.js . . . ?



**そもそも作る  
システムは  
サーバー側が必要か、  
外部サービスで代替で  
きないか**

# バックエンドの必要性を考える

## ・インフラ選定

- AWS/EC2などは手軽と言えども、IaaSはやらないといけない範囲は結構ある
- HerokuやGAEなどのPaaSでも世の中の90%のWebサービスは作れる(らしい)
- ミニマムで始めて、ユーザーがついてスケールが必要になったら差し替えれば良い

## ・サーバーサイド

- サーバーサイドレンダリングが必要なもののか
- ユーザーのデータの管理は自分たちで行う必要があるか
- 静的サイトで代用できはしないか

# フロントエンドの必要性を考える

- アニメーションはマストか
- JS実装がなくてもリリースはできないか(HTML, CSSのみ)
- Web表示速度やSEOの方が優先されないか
- どのページから順番に作るか
  - どのページを自分たちや外部の人はシェアをするのか
  - 導線はどうなっているか
- とはいえ、Web開発でHTML,CSS,JSはマストなので管理をどうするか

# 運用時を考えつつも最速で

- フロント側は差し替え=ユーザー側に影響が出やすいため初期の設計は重要
- バックエンドや運用方法などはある程度差し替えがきく
- URLやドメインだけはどんなにスピード重視でも気をつけたほうがいい



# サーバーレスアーキテクチャ

- 最近のトレンドっぽい
- 自分でバックエンドを用意しないで、なるべく外部サービスに任せるアーキテクチャ

# この辺のツールが使える

- 決済: PayPal, Stripe
- 認証: Auth0, AuthRocket
- ホスティング: GitHub Pages, Heroku, Google App Engine, netlify
- SSL: CloudFlare
- ストレージ、リアルタイム通信:
  - Milkcocoa, FireBase, PubNub
- Webpush: sendpluse
- Webhook: requestBin

**探すと結構いろいろなツールがあるので  
選択肢を多く知っておく  
(RSSでニュースは読もう)  
ことが重要です**

**探すと結構いろいろなツールがあるので  
選択肢を多く知っておく  
(RSSでニュースは読もう)  
ことが重要です**

# ここでのまとめ

- ・ 開発の速度は重要だけど設計もある程度は重要
- ・ ユーザーに近い所ほど最初に設計をしましょう
- ・ 今(時間をかけて)作ろうとしている機能は**本当に今必要なのか、自分で作る必要があるのか**を考えると良い
- ・ 。 。 。 あとは好きな技術や使ってみたいってのも、モチベーション維持的にはある程度必要な気がします  
(小声)

# 今日のまとめ

- ・ 広い視野から見るとエンジニアリングが最優先ではない状況が往々にしてありえる
- ・ ビジネスとクリエイティブの両立を目指すならばラビットプロトタイピングで最速で改善できる仕組みを作しましょう
- ・ チームの状況を冷静にみて判断しましょう