

# フロントエンドの スキルマップと 育成のはなし

Ayumu Sato

June 21, 2014 Frontrend In Nagoya



さとう

あゆむ

佐藤 歩



ハンドルネーム

@ahomu



年間維持費 ￥8,480

<http://aho.mu>

# Job

株式会社サイバーエージェント

アメーバ事業本部

Web（フロントエンド）エンジニア

# Agenda

フロントエンドエンジニアって…

新卒ケーススタディ

「専門」と「分業」と「生存」

さいごに




フロントエンド  
エンジニア



フロントエンド  
エンジニア

って何？

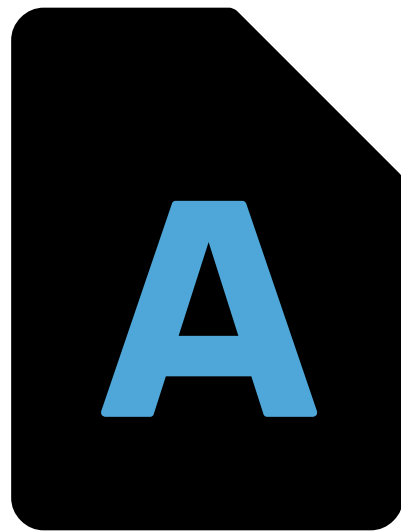




なりたい？  
雇いたい？  
名乗りたい？

具体的に  
なにをする人？





HTML



CSS



JavaScript

Browser

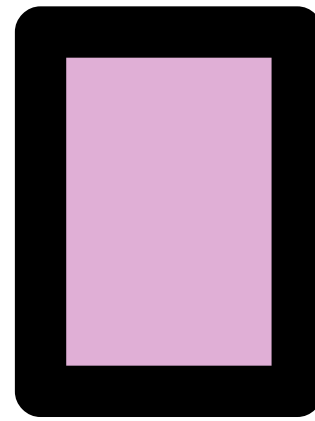




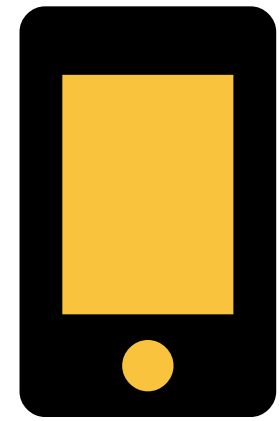
Desktop



Laptop



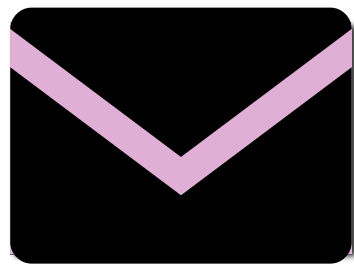
Tablet



Mobile



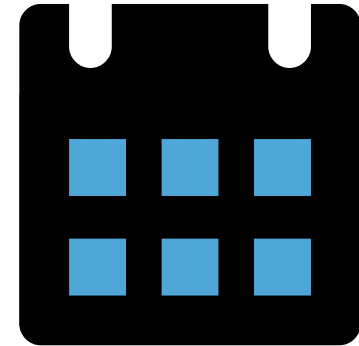
Homepage?



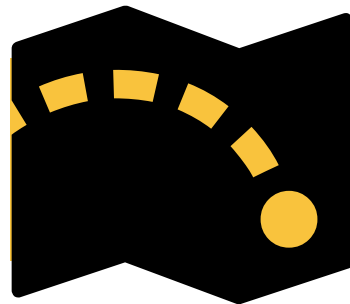
Mail



Shopping



Schedule



Map



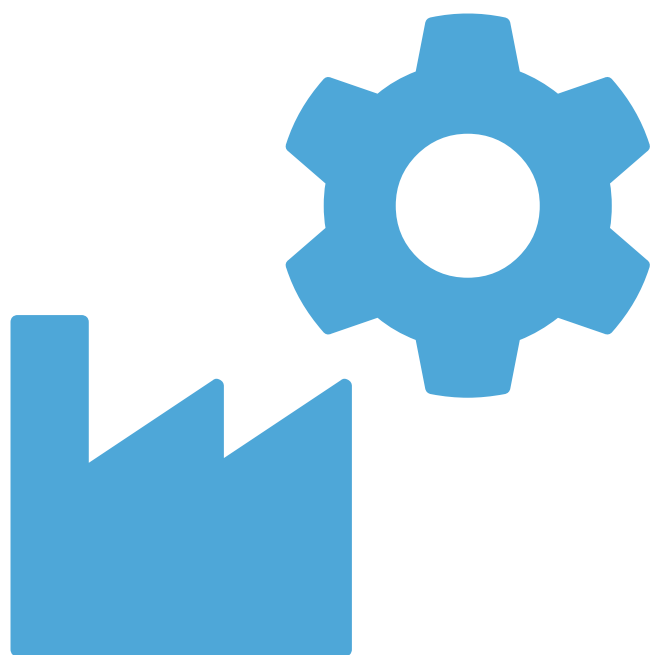
Cloud



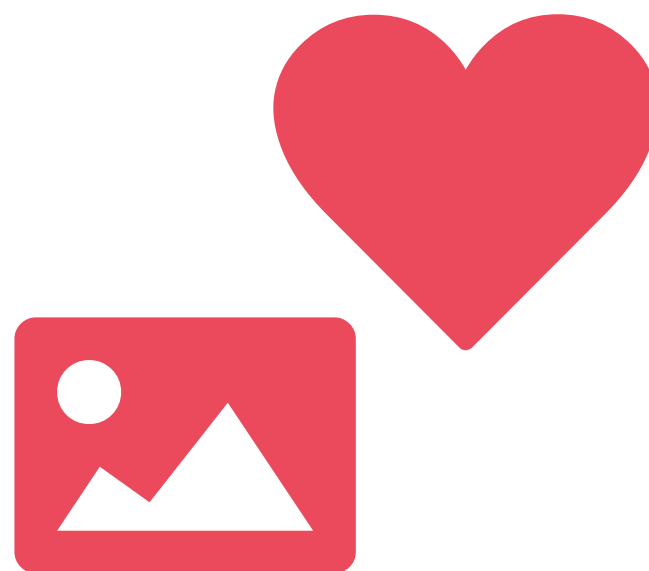
# 色々な役割がある

---

- ▶ Webサイトのマークアップをするひと
- ▶ 複雑なJavaScriptの設計が得意なひと
- ▶ Ajaxを駆使した機能を実装するひと
- ▶ ゲームのアニメーションを開発するひと
- ▶ etc...



エンジニアリング



デザイン

# FEのデザイン

---

- ▶ こういうラベリングだと伝わるかな？
- ▶ 今のUIは操作しづらいから、もっと良くしよう
- ▶ この余白の取り方なら解像度が変わっても大丈夫
- ▶ 気持ちのよいアニメーションはこんなもんかな
- ▶ UIの規則性がないから整理しよう

# FEのエンジニアリング

---

- ▶ ここを自動化したら作業効率があがる
- ▶ 通信回数を減らして読み込みを速くしよう
- ▶ もっと処理の計算量を減らせるな
- ▶ ModelとViewをああしてこうして...
- ▶ データバインディングがどーのこーの

# FEの専門性（例）

---

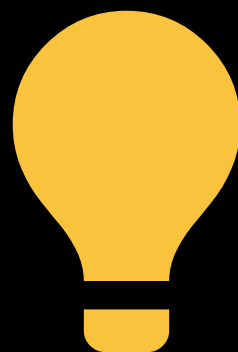
- ▶ ブラウザ・実行環境のパフォーマンスとかを深く理解しているのはフロント
- ▶ ブラウザ依存でバグっても笑ってさっさと対応できるのがフロント
- ▶ CSSのような記述言語に設計を持ち込んでメンテナビリティを保つのもフロント
- ▶ サーバーサイドと連携して複雑な機能と UI を構築できるのもフロント
- ▶ デザイナーと連携してUIやビジュアルデザインを実装に落とし込めるのもフロント

# FEっぽさのまとめ

---

- ▶ ブラウザで動作する技術全般を駆使できる
- ▶ 環境の多様化に追従できる
- ▶ UI・デザイン・インタラクションの実装に責任をもてる
- ▶ エンジニアリングの観点で効率化やパフォーマンス向上を考えられる

新卒ケーススタディ





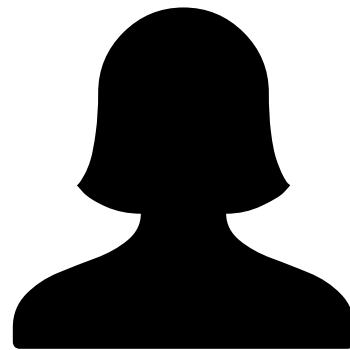
若手の1年(例)

# 若手N

---

技術者としては普通っぽい新卒  
企画・UI の方面に目が良い

時代が時代なら実装もできるよ系の  
Webデザイナーになってたかも？



# 6月

---

- ▶ 配属 & 追加研修
- ▶ NodeとかPhantomJSなど
- ▶ JSだけどブラウザじゃないのを中心に
- ▶ Webアプリが動くための流れを捉えてもらう

自分で考えたモノを  
粗くても作れるように

# 7月

---

- ▶ エンジニア文化の刷り込み
- ▶ 黒い画面、こわいとか言わせない
- ▶ Gitバージョン管理・Vagrant+Chef環境構築
- ▶ 少しずつ開発作業も始める
- ▶ デバッグと問題解決のトレーニングも

刷り込み大事  
何も分からないうちこそ  
狙えるタイミング

# 8-9月

---

- ▶ リニューアルに伴う新規開発
- ▶ 工数を管理してあげながら  
黙々と業務レベルを上げてもらう
- ▶ 目の前の課題に、素直に必死こいてOKな時期
- ▶ この時期にタスクの自己管理まで並行するのは  
個人差あるだろうけど難しい印象

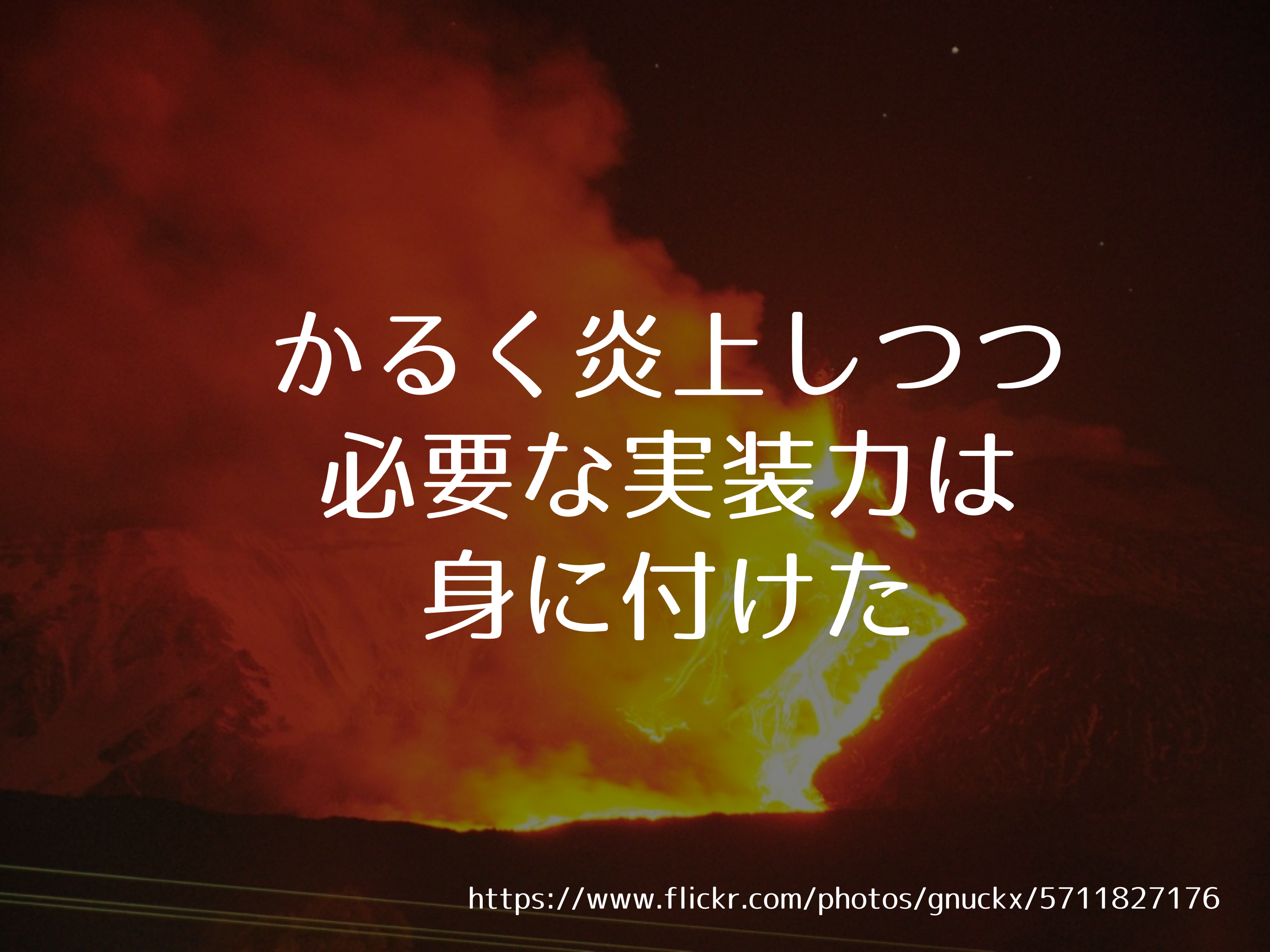


フロントの実装に  
必要な前提技術を  
しっかり抑える

# 10-11月

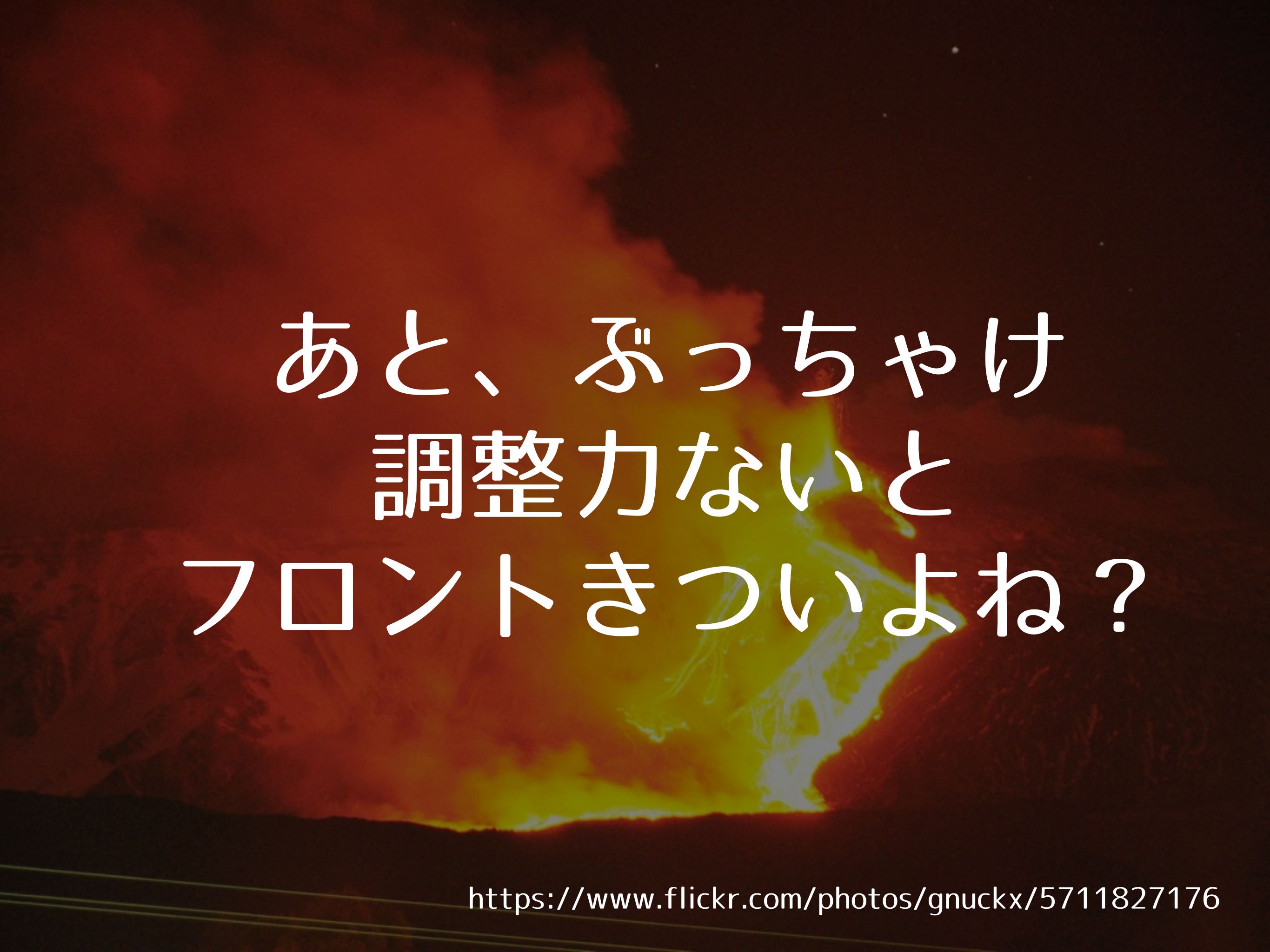
---

- ▶ リリース後の施策実装
- ▶ 通称「鬼の運用」
- ▶ リリース前よりもスピード感が大事
- ▶ 正確な見積もりと、手早い実装が求められる

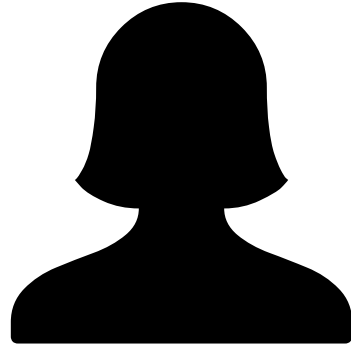


かるく炎上しつつ  
必要な実装力は  
身に付けた

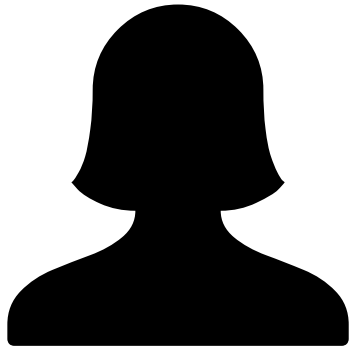




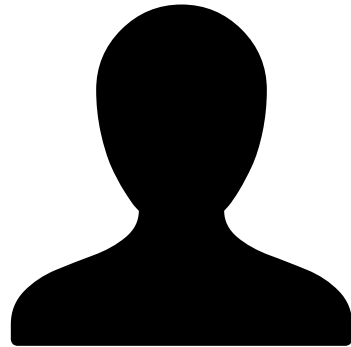
あと、ふっちゃけ  
調整力ないと  
フロントきついよね？



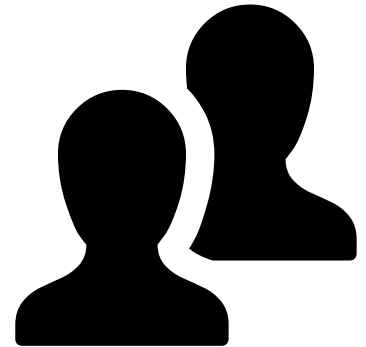
プロデューサー



デザイナー



フロントエンド  
エンジニア



サーバー  
エンジニア

# 12-翌1月

---

- ▶ リリース管理のハンドリング、調整をお願いする
- ▶ もちろん実装もやる
- ▶ ついでに自分が多忙になる（強制親離れ）
- ▶ ちょっと死にそう

以降、独り立ち



# ふりかえり

---

- ▶ 自分で考えたモノを粗くても  
ひとりで作れるように
- ▶ 何も分からないうちに、当たり前を刷り込む
- ▶ 実装は前提技術が揃っていれば、根性で出来る
- ▶ 適切な見積もりと、チーム内の調整力を備える

# 残された課題

---

- ▶ フレームワーク的なものは使えるがスクラッチのアーキテクチャが組めない
- ▶ 複雑なUIを混乱せずに実装しきる力が弱め
- ▶ と、思ったら File API と Canvas を使った画像トリムの UI とか作ってたので褒めた
- ▶ ひとは勝手に / (^o^) \ 成長するんだ！

フロントエンドの  
スキルを覚える？

HTML

CSS

JavaScript

Ajax

HTTP

WebSocket

Java

PHP

Node

npm

grunt/gulp

shell

UI

Interaction

RWD

Visual

Typography

Vertical Rhythm

HTML

CSS

JavaScript

Ajax

HTTP

WebSocket

Java

PHP

Node

エトセトラ...

npm

grunt/gulp

shell

UI

Interaction

RWD

Visual

Typography

Vertical Rhythm

全部おぼえるのは  
無理/困難

1から10まで教えるのは  
もっと困難

本人が学び方を  
身に付ける



序盤が特に重要

学びの習慣と手段を  
身に付けてもらう

# 学ぶ習慣

---

- ▶ フログ
- ▶ 勉強会
- ▶ 書籍
- ▶ ライバル探し
- ▶ メンター探し
- ▶ Googleる
- ▶ Referenceる
- ▶ Stack Overflowる
- ▶ Qiitaる
- ▶ GitHubる

# 学ぶ習慣

---

▶ フログ

▶ 勉強会

▶ 書籍

▶ ライバル探し

▶ メンター探し

▶ Googleる

▶ Referenceる

▶ Stack Overflowる

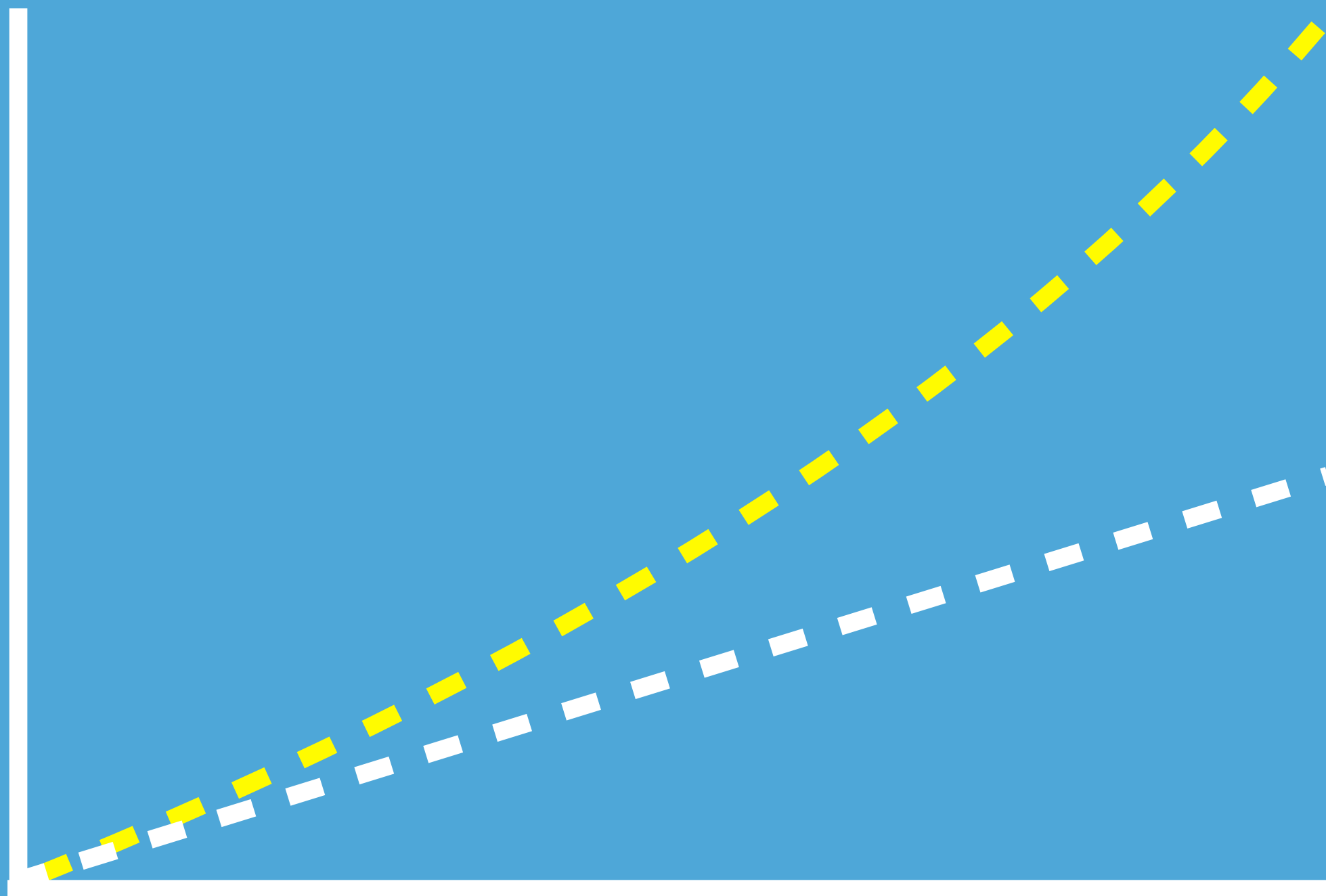
▶ Qiitaる

▶ GitHubる



学びも複利が大事

成長



時間





学ぶ習慣さえあれば  
ちよつと分野が  
変わっても大丈夫

# 外から働きかける 学習支援

\* 独学の支援 \*

# 外から働きかけられること

---

- ▶ ギャップの割り出し
- ▶ ゴールと機会づくり
- ▶ 適度なフィードバック
- ▶ 評価（よくやった！）

# ティーチングと コーチングの 使い分け

最初はティーチング  
しっかり基礎を教える

慣れたらコーチング  
本人の学びを引き出す

# 質問の仕方

「どうしたらいいですか？」



「どうしたら良いか  
自分なりに答えを  
出してから聞いてね」

「答えを教えて貰うより  
自分で考えてから  
答え合わせを聞いたほうが  
勉強になるでしょう？」

# 弊社取締役 人事本部長の言



たとえば営業で持参資料を準備するのに  
自分で考えて準備するのと  
先輩に指示された通りに準備するのとで  
は大きな差が生じます。

<http://www.executive-interview.com/archives/021/>

— 人 人 人 人 人 人 人 —  
> 決断経験値 <  
— Y ^ Y ^ Y ^ Y ^ Y ^ Y —

# 外から働きかけられること

---

- ▶ ニーズの割り出し（現状評価）
- ▶ 機会づくり（ゴール設定）
- ▶ 適度なフィードバック
- ▶ 評価（よくやった！）

# 効率良く学ぶための 学習設計

# インストラクショナル デザイン

のエッセンス



何の前提を満たせば  
課題をクリアできるか

# \* 課題分析 \*

DOMイベントの  
知識

touchイベント  
の制御

jQueryの  
使い方

DOM操作の  
やりかた

カルーセルの  
実装

アニメーション  
の制御

CSSの知識

こういうのを踏まえて  
ゴール作ると良いハズ

本人の学びと  
学習支援が肝

「専門」

と

「分業」

と

「生存」



A person wearing a blue Superman costume with the iconic red and yellow 'S' shield on the chest. They are holding two large, silver combat knives, one in each hand, with the blades pointing outwards. The background is dark and out of focus.

# フルスタック エンジニア 最強伝説

<https://www.flickr.com/photos/xurbler/376591423>

フルスタック  
エンジニア



# フルスタック エンジニア

って何?

具体的に  
なにをする人？



分業の変化は  
少なからずある

# 開発工程の水平分業

---



インフラ



デザイン



マークアップ

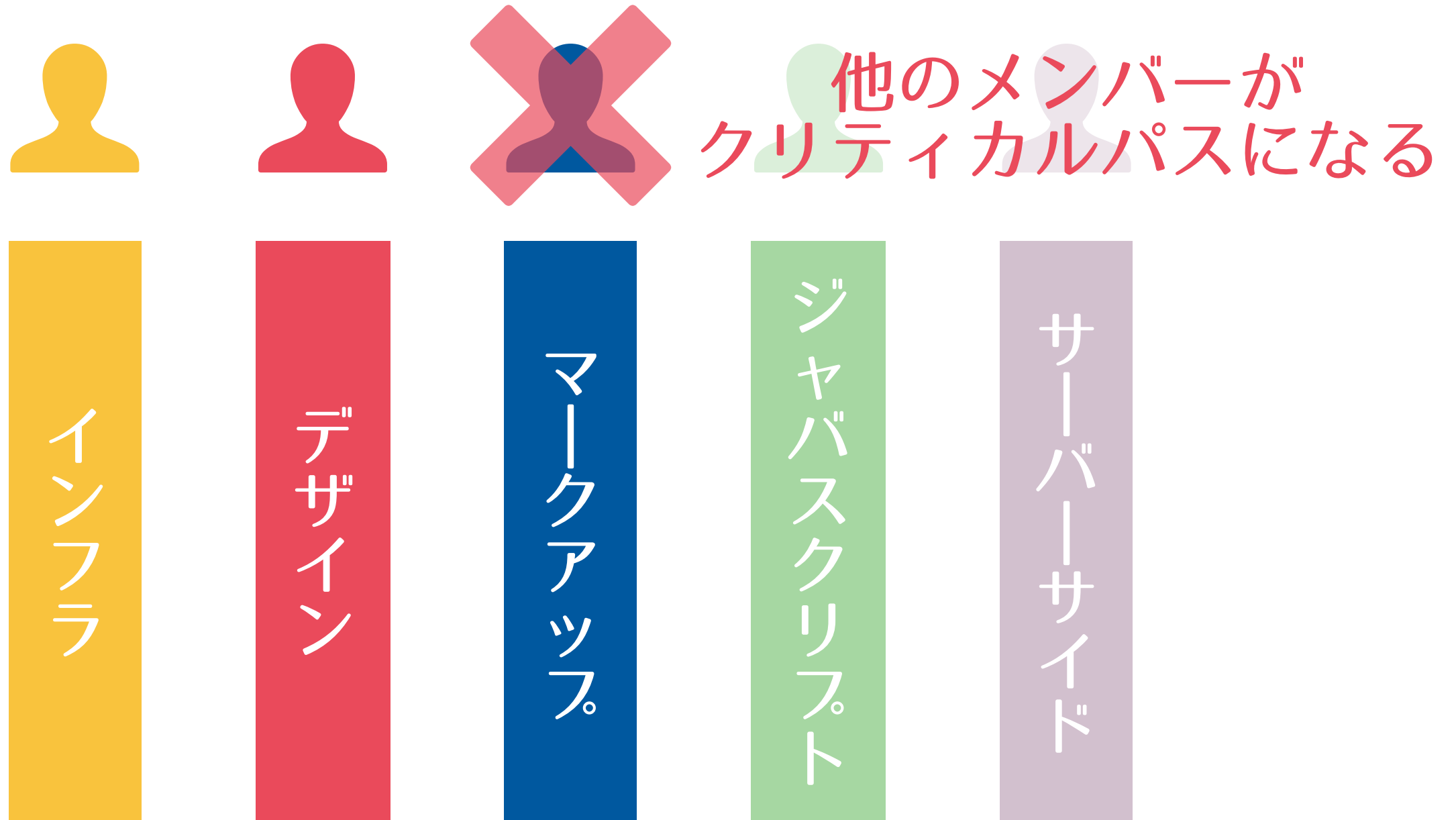


ジャバスクリプト



サーバーサイド

# 開発工程の水平分業



# 機能単位の垂直分業



サーバー  
サイド

ジャバ  
スクリプト

マーク  
アップ



サーバー  
サイド

ジャバ  
スクリプト

マーク  
アップ



サーバー  
サイド

ジャバ  
スクリプト

マーク  
アップ

デザイン

インフラ



# 機能単位の垂直分業



サーバー  
サイド

ジャバ  
スクリプト

マーク  
アップ



サーバー  
サイド

ジャバ  
スクリプト

マーク  
アップ



サーバー  
サイド

ジャバ  
スクリプト

マーク  
アップ

デザイン

インフラ

# スペシャリスト と ゼネラリスト



Web技術者の名乗りは  
インフレし続ける

ひとつひとつの事が  
簡単になっている

インフラはPaaSあるし  
サーバーはフレームワークあるし  
フロントもライブラリあるし

中長期的に  
皆が何でもできる  
という日は来てしまう

今のフロントエンドも  
—昔前は特殊技能に  
聞こえた

でも今や  
フロントエンドなんて  
沢山いませんか？

フロントエンドで  
何をしてるの？

フルスタックで  
何をしてるの？



最終的には  
何ができるひとなのか？

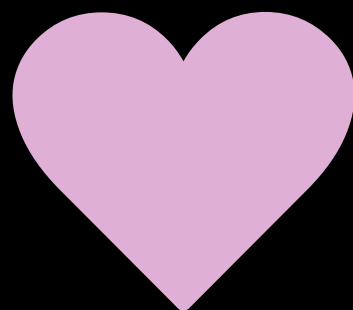
素晴らしいUIを  
提案するひと？

ブラウザゲームを  
作るひと？

高速でWebサービスを  
立ち上げる人？

何をする人なのか  
ブランディング

さいごに





フロントエンド

VS

サーバーサイド



クライアントサイド

VS

サーバーサイド



ネイティブのほうが  
近いかも

クライアント技術は  
変わりやすい！

Webの将来的な技術で  
ワークフローが  
変化することで？

例えば Web Components

コンポーネント  
デザイナー

コンポーネント  
ディベロッパー

# コンポーネント ビルダー

やりたいことで「肩書き」が決まる  
ほしい人材で「募集職種」が決まる



「肩書き」や「募集職種」は  
時代に依存して  
必ず変化がやってくる

# Questions?

🏠 <http://aho.mu>

🐦 [@ahomu](https://twitter.com/ahomu)

🐙 [github.com/ahomu](https://github.com/ahomu)

<https://www.flickr.com/photos/futurilla/5799045658/>

