

# フリーランスエンジニアが 受託案件をスムーズに 進めるために心がけてること

~ DIST.19「フリーランスの生存戦略」~



## Go Nishiduka

@resistance\_gowy

Front-end engineer

⌚ 東京都

🔗 [htmlgo.site](http://htmlgo.site)

あとらす二十一 → SONICJAM → フリーランス

主にウェブサイト作ってます。

受託案件をスムーズに進めるために  
普段心がけていること、  
「なるべく効率よく、他のメンバーにもや  
さしい実装方法」のお話。

# Webサービス系企業の場合

- ・チーム内でコーディング規約があり、メンバーがそれを守って開発を行う
- ・コードレビューを行なって品質を守っている
- ・毎日MTGを行い、情報や進捗を共有、コミュニケーション方法も統一
- ・基本的に同じメンバー間で開発を行う

すごくスマートな環境！

# 受託案件(フリーランス)の場合

- ・案件毎にルール、レギュレーションが違う、無い。やばいコードに出会うことも。
- ・メンバーも毎回新しい人とお仕事をやる、スキルもまちまち。
- ・コミュニケーションがメール、鬼電、修正ppt送りつけ、、、
- ・クライアントまで何層も人を介したりして、情報の伝達が悪い。

すごくカオス！

メンバーでの仕事の進め方、意思の統一が出来て  
いるかでその案件のスムーズ度が違う。  
そして、案件ごとに毎回その土台作りが必要。

しかし、受託の特性上 全てをスマートな環境にするの  
は不可能な話なので、

出来る限り、スムーズに運べるようにする。

# コミュニケーション編

コミュニケーション方法はかなり重要。  
ここをきちんと整備出来るかで  
やりとりの工数や、ストレスが変わってくる。

とある作業の依頼メール・・・

hoge様

お疲れ様です。  
hugaです。

最新のスケジュールとワイヤーフレームを作成いたしましたので添付いたします。

また、修正をお願いします。

- ・KVの画像を最新のものに差し替え
- ・リストの箇所を15件でランダム表示
- ・○○さんページのリンク先を [hogehoge.jp](http://hogehoge.jp)に変更
- ・アニメーションのeasingをもうすこし抑揚をつける

こちらの作業 なるはやで対応お願いします。

お見積もりも、今週中にいただけるとありがとうございます。

よろしくお願いします。

---

## 2 個の添付ファイル



hoge様

お疲れ様です。  
hugaです。

最新のスケジュールとワイヤーフレームを作成いたしましたので添付いたします。

また、修正をお願いします。

- ・KVの画像を最新のものに差し替え
- ・リストの箇所を15件でランダム表示
- ・○○さんページのリンク先を [hogehoge.jp](http://hogehoge.jp)に変更
- ・アニメーションのeasingをもうすこし抑揚をつける

こちらの作業 なるはやで対応お願いします。

お見積もりも、今週中にいただけるとありがとうございます。

よろしくお願いします。

## 2 個の添付ファイル



# 問題点

- ・メールでのやりとりだと件名が「Re:Re:Re:修正の依頼」のようになり、あとからメールの内容を検索しづらい、そもそも何の作業依頼なのかが不明。
- ・1つのメールで3つの依頼が来てしまっている。  
さらにその中で複数の修正依頼があり、漏れる可能性がある。優先度も謎。
- ・更新する可能性のあるファイルが添付されているので、色んなバージョンのものが散る可能性がある。



# ヌーラボ社が運営している プロジェクト管理ツール

**Backlog**  
by nulab

Backlogの特長 プランと料金 チームでの利用例 ▾ サポート 無料トライアル

## チームではなく、すべての人に

Backlogはウェブ制作、ソフトウェア開発、大手広告代理店、全国版新聞社など様々な業種で使われているプロジェクト管理ツールです。

[無料トライアル](#)



このツールを使用すると  
先ほどの例は

GO testdesu (TESTDESU)



△ 検索条件 シンプルな検索 高度な検索

@ 短いURL

検索条件を保存

状態: すべて 未対応 处理中 处理済み 完了 完了以外

親子課題: すべて 親課題 子課題以外

カテゴリー

マイルストーン

担当者

キーワード

すべて

すべて

すべて



キーワードを入力

全 6 件中 1 件 ~ 6 件を表示 1

まとめ操作

表示設定

...

種別	キー	件名	担当者	状態	優先度	登録日	期限日	更新日	登録者	添付
タスク	TESTDESU-5	アニメーションのeasingをもうすこし抑揚をつける	GO_NISHIDUKA	完了	→	2018/01/28		2018/02/01	GO_NISHIDUKA	
タスク	TESTDESU-2	KVの画像を最新のものに差し替え	GO_NISHIDUKA	処理済み	→	2018/01/28		2018/02/01	GO_NISHIDUKA	
タスク	TESTDESU-3	リストの箇所を15件でランダム表示	GO_NISHIDUKA	処理済み	→	2018/01/28		2018/02/01	GO_NISHIDUKA	
タスク	TESTDESU-6	見積もり作成	taro	未対応	↑	2018/01/28	2018/02/09	2018/01/28	GO_NISHIDUKA	
▶ タスク	TESTDESU-1	hogehogeページ修正	(1/4) GO_NISHIDUKA	未対応	→	2018/01/28	2018/02/22	2018/01/28	GO_NISHIDUKA	
タスク	TESTDESU-4	○○さんページのリンク先を hogehoge.jpに変更	GO_NISHIDUKA	未対応	→	2018/01/28		2018/01/28	GO_NISHIDUKA	

全 6 件中 1 件 ~ 6 件を表示 1

まとめ操作

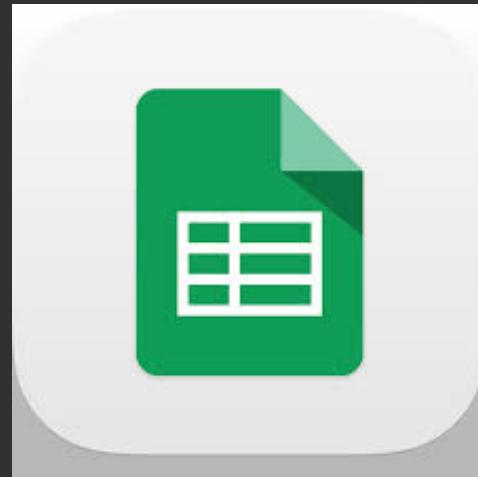
表示設定

...

# 改善点

- ・タスクが細かい粒子でリスト化されて、残っているタスクが一目瞭然(子課題もたてられる)。終わったらタスクを完了にして、全部無くなったら終了。
- ・作業担当者、タスクのステータス、優先度が可視化された。
- ・タスクごとにスレットが立てられたので、他のタスクとやりとりが混ざらなくなった。

# スケジュール ワイヤーフレーム



出来れば

CACOOや、Googleスプレッドシートなどのオンラインツール  
を使って一元管理してもらうのが理想。

でも、クライアントやディレクターに強制するのは  
難しいので、(セキュリティ的に無理です、等言われることも)  
その辺は適宜。

何だかんだで現実的なライン、  
Officeのファイルで管理して、  
Backlogのフォルダに日付ごとにファイルを格納

メールで無限に添付してくると、どれが最新か  
わからぬのでやめましょう！（やめて）

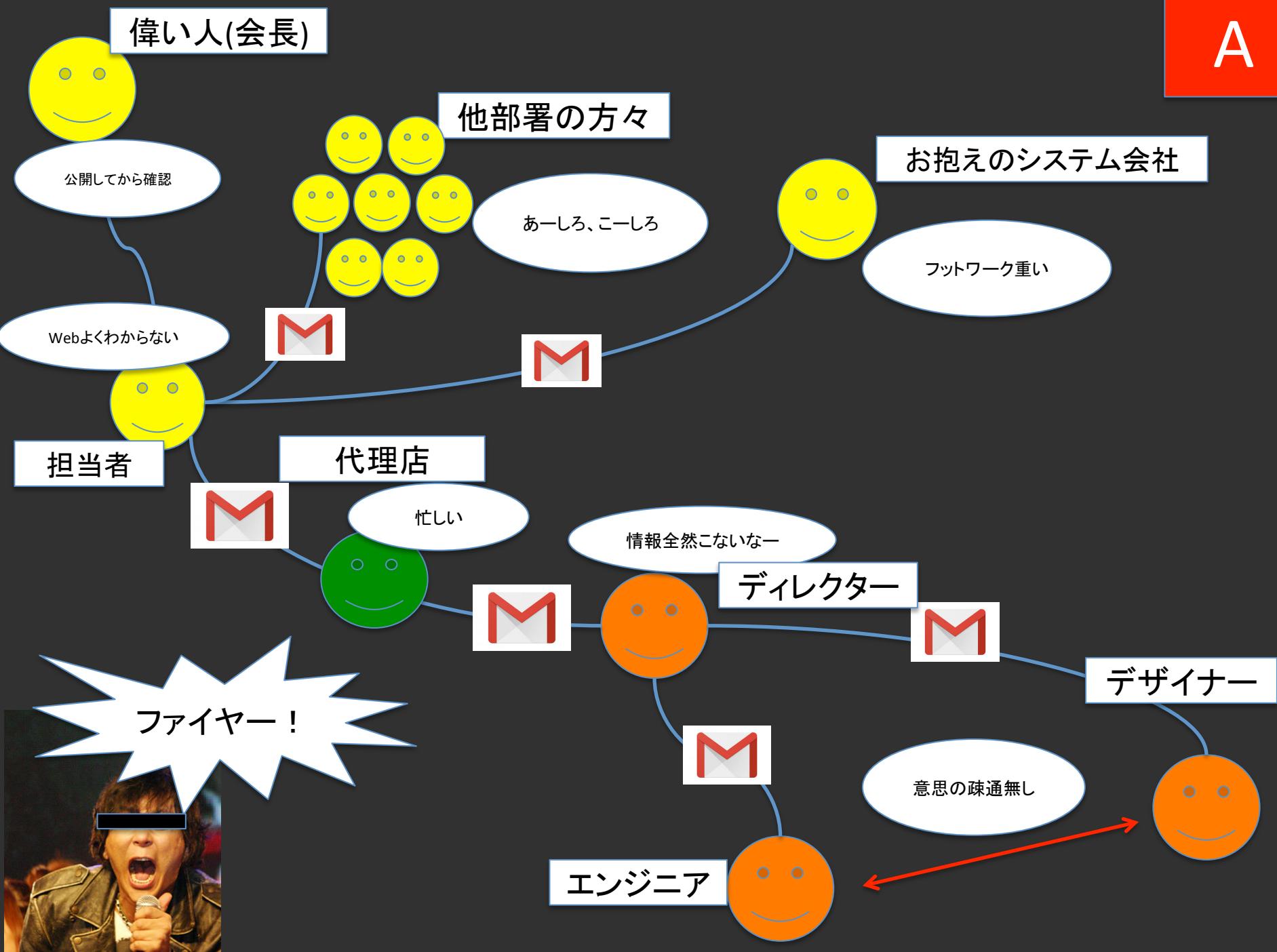
ファイル

新しいフォルダ 新しいファイル ブラウザで開く

<input type="checkbox"/> すべて選択	チェックしたものを	<a href="#">移動する</a>	<a href="#">削除する</a>	<a href="#">ダウンロードする</a>
名前	サイズ	更新者	更新日時	
<input type="checkbox"/> スケジュール	-	GO_NISHIDUKA	2018/01/28 16:53	
<input type="checkbox"/> ワイヤーフレーム	-	GO_NISHIDUKA	2018/01/28 16:52	
<input type="checkbox"/> すべて選択	チェックしたものを	<a href="#">移動する</a>	<a href="#">削除する</a>	<a href="#">ダウンロードする</a>

# 燃える案件の座組み (ちょっと盛ってます)

A



Backlogを  
クライアントにも  
使ってもらって  
円滑パターン

B

クライアント

システム

部長

担当者

制作

エンジニア

ディレクター

デザイナー



# やりとりのコストがスムーズ！ 理想形

ただし、ルールを決めてあげないと  
割とめちゃくちゃになりがちなので、

- ・キックオフMTGの際に使い方の共有
- ・タスクの雛形を作ってあげる

Backlogを  
制作サイドのみで  
使うパターン  
(大体これが多め)

C

クライアント

システム

部長

担当者

制作

ディレクター

エンジニア

デザイナー

backlog  
by nulab



# 進行はスムーズ

これだと、ディレクターの負荷は高い、  
タスク登録しなおしたり、  
技術的なやりとりに齟齬が生まれたりする。

- ・専門的なやりとりはクライアントと直で
- ・なるべくディレクターに協力する
- ・タスクは制作サイドみんなで管理

C`

クライアント

システム

部長

担当者

制作

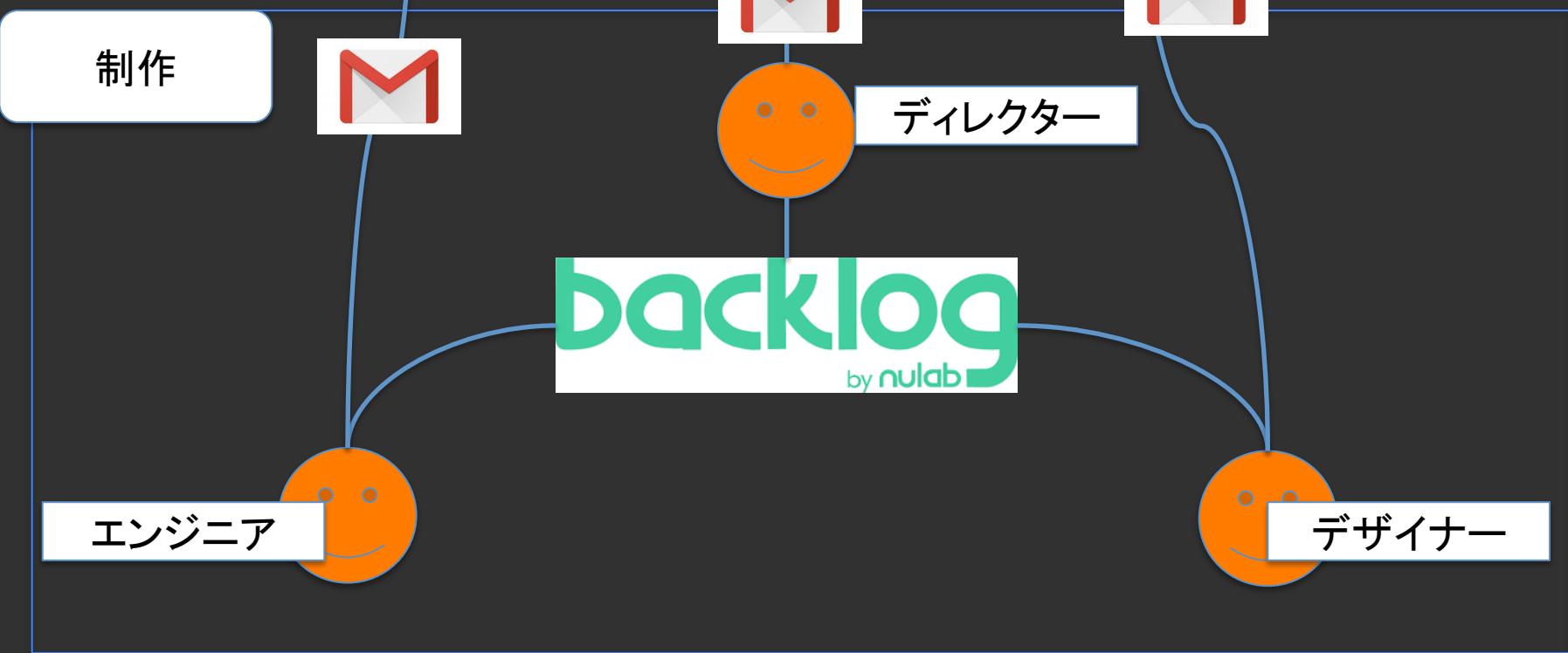


ディレクター

エンジニア

デザイナー

backlog  
by nulab



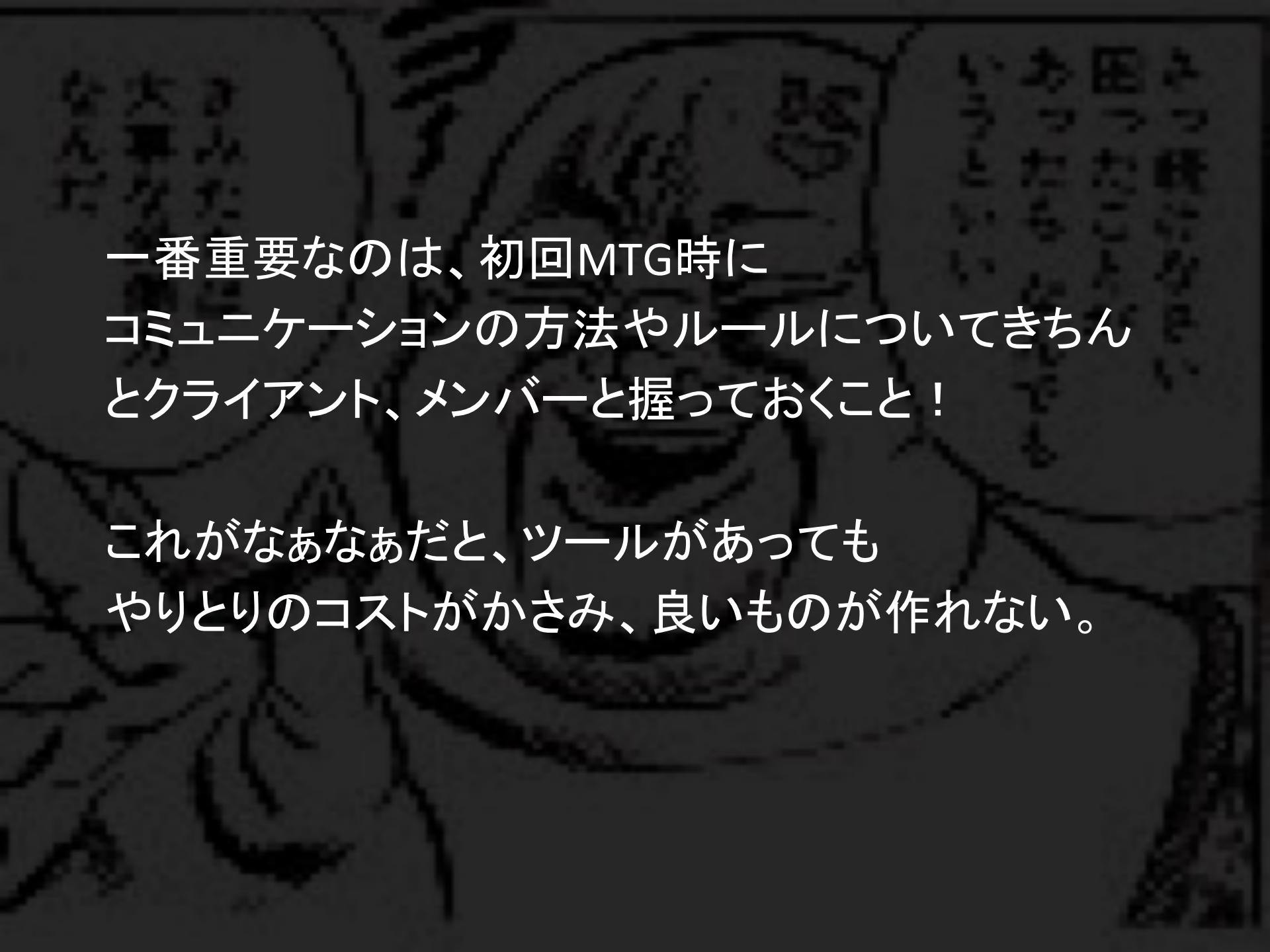
あとは、エンジニアに関して言えば、

- ・モヒカンをやめる

→エンジニアが「それわかんないでしょ…」、っていうようなワードを羅列して、ディレクターが「…」ってなってるときがあるので、噛み砕いて説明してあげる。

- ・文章を作つてあげる

→技術的、専門的なやりとりをメールで送るとき、ディレクターにコピペでそのまま送れるように文章を作つてあげる。可能なら直でやりとりする。



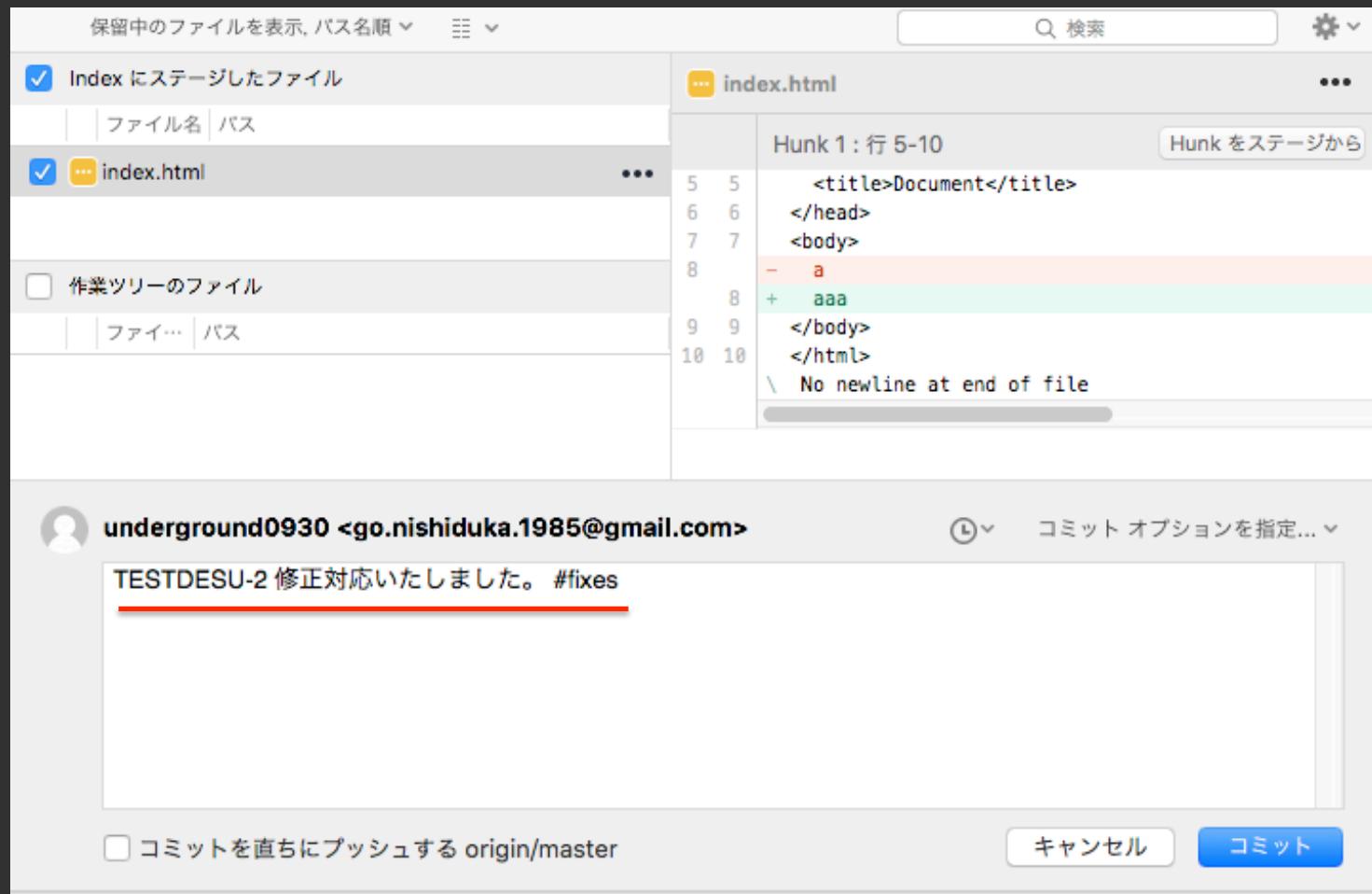
一番重要なのは、初回MTG時に  
コミュニケーションの方法やルールについてきちんと  
クライアント、メンバーと握っておくこと！

これがなあなあだと、ツールがあっても  
やりとりのコストがかさみ、良いものが作れない。

# ちなみに、backlogの便利な機能

- ・gitが内蔵されていて、タスクとコミットを連動出来る。
- ・wikiをpdfにしてダウンロード出来る
- ・他もあるけど今回は割愛。

# 内蔵されたgitに適切なコメントでpushすると、、、



タスクID

コメント

ステータス変更

TESTDESU-2 修正対応いたしました。 #fixes



コメント (1)

表



GO\_NISHIDUKA

2018/02/01 14:18:06

- ➡ ○ 状態: 未対応 → 処理済み
- コミット: [d17124b906](#) ( test(TESTDESU) の master にプッシュ )

➡ TESTDESU-2 修正対応いたしました。 ←

# Wikiに仕様などを細かく書いておいてPDFにして

クライアントでのバナー更新方法

以下の配列を修正すると、  
vol1~vol5のbacknumberのバナーの  
日本語を編集できます。

```
var bannerArray = [  
    'ああああああああああああ',  
    '第2弾のタイトル入ります',  
    '第3弾のタイトル入ります',  
    '第4弾のタイトル入ります',  
    '第5弾のタイトル入ります'  
];
```

編集 ファイルを添付 ...

共有ファイルをリンク  
PDF  
履歴  
コピーを作成

## ●headerのページ送り

それぞれのバージョンに最適化されます。  
例えば、vol3までの公開なら数字に3を入れてください。  
以下に、数字ごとの各ページのリンク先をまとめます。

```
launchPage = 1
```

現在のページ	前ヘリンク	次ヘリンク
vol1	無し	無し

file:///TESTDESK/262224/クライアントでのバナー更新方法.pdf

# このままクライアントに展開。

以下の配列を修正すると、  
vol1~vol5のbacknumberのバナーの  
日本語を編集できます。

```
var bannerArray = [  
    'ああああああああああああ',  
    '第2弾のタイトルあります',  
    '第3弾のタイトルあります',  
    '第4弾のタイトルあります',  
    '第5弾のタイトルあります'  
];
```

## ●headerのページ送り

それぞれのバージョンに最適化されます。  
例えば、vol3までの公開なら数字に3を入れてください。  
以下に、数字ごとの各ページのリンク先をまとめます。

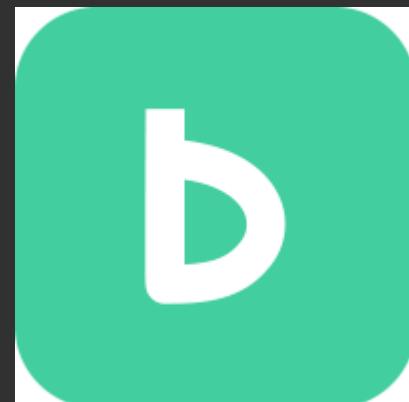
launchPage = 1

現在のページ	前ヘリンク	次ヘリンク
vol1	無し	無し

launchPage = 2

現在のページ	前ヘリンク	次ヘリンク
vol1	無し	vol2
vol2	vol1	無し

オールインワンで便利！！



# 実装編

# 80点を目指す

ここにおける80点の意味とは、

- ・個人の開発のしやすさだけを追求しない。
- ・なるべくシェアが高い技術を使う。



# HTML テンプレートエンジン の選定

# 人気どころでいうと下記3つ



<%= EJS %>





- javascriptが使える。
- include とか extendが便利
- インデントベースでコード量が少ないので開発が早い
- 開発が活発でシェアもある
- 複雑に書くと、コードが読みづらい
- htmlとの親和性がよくないのでコピペしてくる、などは厳しい。
- 学習コストは他に比べて高め。

```
doctype html
html(lang="en")
  head
    title= pageTitle
    script(type='text/javascript').
      if (foo) bar(1 + 5)
  body
    h1 Pug - node template engine
    #container.col
      if youAreUsingPug
        p You are amazing
```

# <%= EJS %>

```
<ul>
  <% users.forEach(function(user){ %>
    <%- include('user/show', {user: user}); %>
  <% } ); %>
</ul>
```

- htmlベース
- includeが便利
- javascriptが使える。
- 開発が活発でシェアもある
- htmlとの親和性が高い。
- 学習コストは低め。

HTML



- ・誰でも触れる
- ・ページの量産がしんどい
- ・SSIで共通化する前提

# (1).公開期間



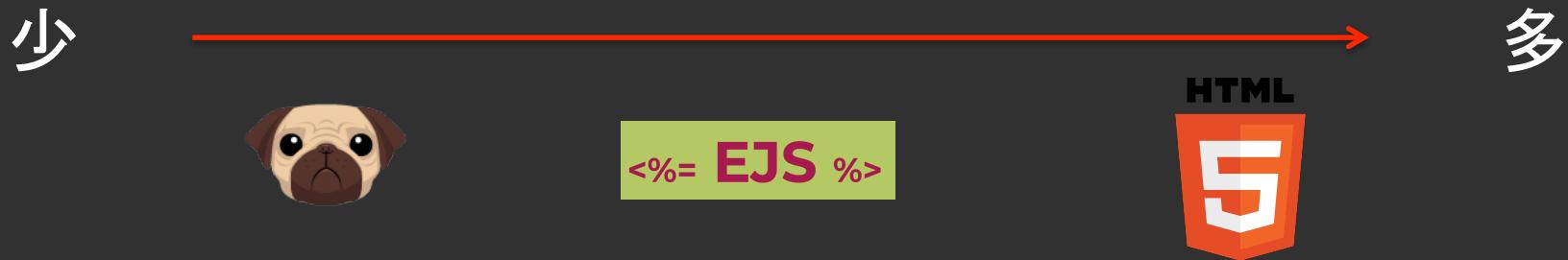
Pug,EJS,は、nodeのバージョン依存や、プラグイン自体の開発が止まる可能性があるので、短い公開期間向き。

## (2).規模、ページ数



インクルードや、javascriptなど機能が充実しているので、  
量産の時は便利。  
ただあまりにファイル数が多くすぎる場合は、htmlの生成  
に時間がかかる可能性がある。

## (3).開発人数



スキルもまちまちだと思うので、人数が多いほどシンプルで難易度が低いものを採用したほうが無難。

# (4). クライアント側で運用する場合

case by case

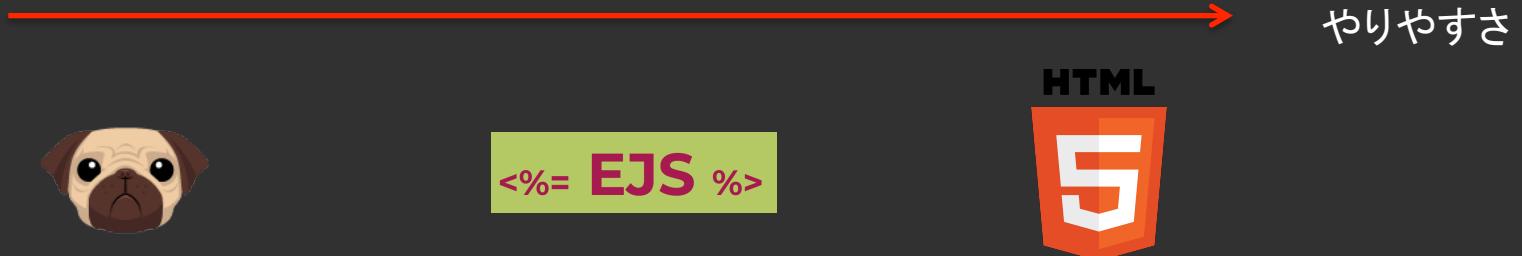


クライアント側で、実装環境が用意できるなら、pug, EJSでも可。

出来ない場合は、納品後は、テンプレートエンジンとは完全に切り離してhtmlベースで運用してもらう。

大きい規模の場合は全部、静的html納品だと運用側がキツイので「html + SSI」の方が良い。

## (5).組み込みがある場合



Wordpress案件など、組み込み側としては、  
Pugで書かれると修正の際にgitでの差分が分かり辛い。

その場合は、ビルトされたhtmlもgitで管理してもらう必要がある。(gitで管理するhtmlの数が倍になる。)

これは完全な愚痴なんですが、  
Gulpとかpugを使う場合は、組み込みサイドとしては  
ビルドされたデータをgitでignoreしないで欲しい！

修正があるたびにpull→タスクランナー走らせる、  
とか狂気の沙汰かと。。  
「Gulpなんて知らない」ってシステム屋もいるわけで。  
それなら直接テンプレ修正して！！と思う。



最近は、フロント側に直接修正してもらいます。逆もし  
かりです。ありがとうございます。

CSS



VS





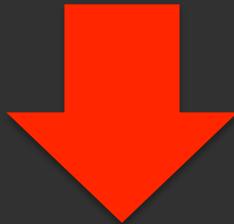
- ・カスタマイズ性に優れていて、沢山のプラグインがある
- ・パフォーマンスが良い
- ・自由度が高いが、カスタマイズされすぎると属人的なコードになってしまう。



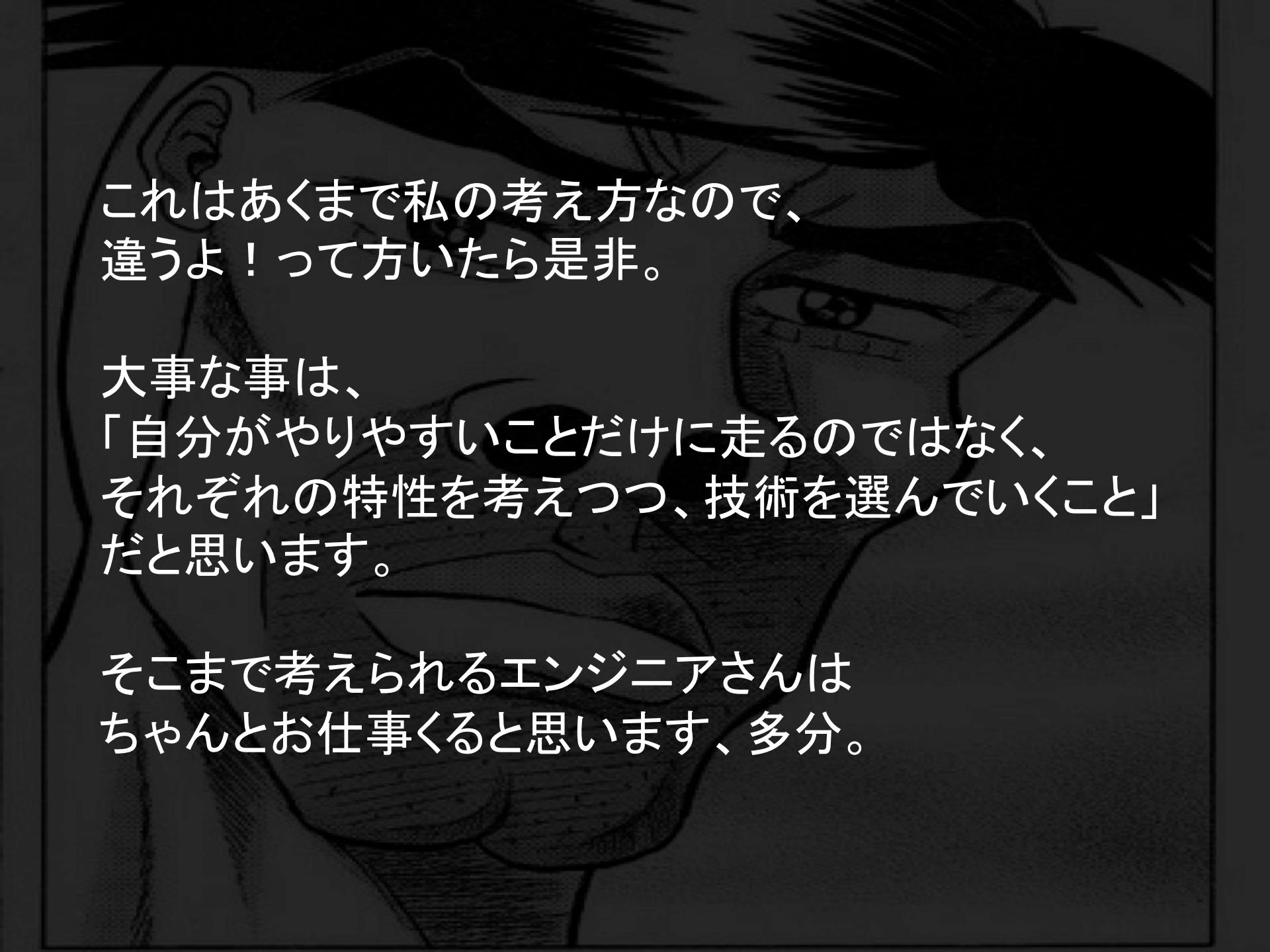
より良いコードを突き詰めていくにはこちらがいいかも。  
限られたメンバー内でCSSを書くことに向いている。



- ・ユーザー数、コミュニティが大きい。
- ・いい意味で枯れている、廃れる可能性は低い。



一通りの便利な機能が揃っていて、ユーザー数が多いので  
不特定多数のメンバーと作業する時に向いている。  
80点目指すならこちらが良い。



これはあくまで私の考え方なので、  
違うよ！って方いたら是非。

大事な事は、  
「自分がやりやすいことだけに走るのではなく、  
それぞれの特性を考えつつ、技術を選んでいくこと」  
だと思います。

そこまで考えられるエンジニアさんは  
ちゃんとお仕事くると思います、多分。

z-index:99999

案件ペンディング！

組み込み後に  
デザイン修正！

なるはや！

IE9で崩れます！

Android  
標準ブラウザ死

スケジュールに連休  
入ってる！

Bemでネスト深すぎ！

インクルードで！

公開後  
会長がやり直しを  
命じる

今日も力オス

おわりです