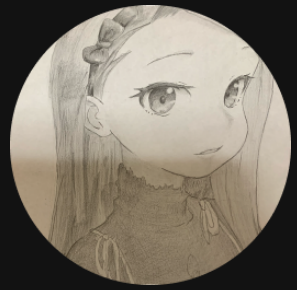
Feature driven folder structure は何を解決する？

2023春大LT

学部 3年 ERASER（加藤 豪）

Press Space for next page →

ERASER



- 加藤 豪
- 会津大学 学部３年
- Twitter, GitHub
- ポートフォリオ

技術

- 言語：JS・TS、CSS、HTML、Lua、Rust(簡単な競プロ)、Haskell(勉強中)
- Webフロント：React.js、Next.js、GraphQL(Apollo Client)、CSS-in-JS
- バックエンド：Node.js、Prisma(ORM)、
- ツール：Git・GitHub、Figma

好きなもの

- Neovim、Wezterm、綺麗なコードを考える
- Vtuber、スプラ、EGOIST

前置き

今回のお話はマサカリの飛びやすい分野だと思います。

もし適当なことを言っていたりしたら、シメていただけると泣いて喜びます。

Feature driven folder structure ってなんぞ？

機能駆動のフォルダ構造です（翻訳しただけ）。

```
src/
└ features/
  ├ login/
    ├ use-login.ts
    ├ login-page.tsx
    └ login-form.tsx
  ├ ui/
    ├ button.tsx
    ├ card.tsx
    └ ...
  └ ...
...
```

Feature driven folder structureは何を解決する？

これが何を解決するのか？
それについて考えていくのが今回のお話です。

こんなReactのコードベース、見たことない？

```
src/
├ components/
├ context/
├ hooks/
├ pages/
├ lib/
└ App.ts
```

よくある、かは知らないが、僕がこれまでよくやっていたReactのコードベース

こんなReactのコードベース、見たことない？

```
src/
├ components/
├ context/
├ hooks/
├ pages/
├ lib/
└ App.ts
```

これは**アンチパターン「技術駆動パッケージング」**の一例です

技術駆動パッケージングというアンチパターン

技術駆動パッケージングというアンチパターン

技術駆動のどこが良くないのでしょうか？

先程のディレクトリ構成の内の一部を中身が見えるようにしてみましょう。

```
src/
├ components/
  ├ login-form.tsx
  └ ...
├ hooks/
  ├ use-login.tsx
  └ ...
└ pages/
  ├ login-page.tsx
  └ ...
...
```

技術駆動パッケージングというアンチパターン

今度は全体ではなく`components`の中を見てみます。

```
src/
├ components/
  ├ button.tsx
  ├ card.tsx
  ├ login-form.tsx
  ├ user-profile-card.tsx
  └ ...
...
```

今回は簡単と誇張のために`components`内部は全てフラットであるものとしました。

技術駆動パッケージングの問題点

まずはこちらから。

```
src/
├ components/
  ├ login-form.tsx
  └ ...
├ hooks/
  ├ use-login.tsx
  └ ...
└ pages/
  ├ login-page.tsx
  └ ...
...
```

- ログインのフックとコンポーネントとページがバラバラの場所にあり、把握がしずらい

技術駆動パッケージングの問題点

そして次にこちら。

```
src/
├ components/
  ├ button.tsx
  ├ card.tsx
  ├ login-form.tsx
  ├ user-profile-card.tsx
  └ ...
...
```

- `button`と`login-form`が同じレイヤにいるなど、`components`の抽象度がバラバラ
- 用途の全く違う`login-form`と`user-profile-card`が同じパッケージ（ディレクトリ）にいる

技術駆動パッケージングの問題点

これらの問題を生み出すものの名前なんというか、我々は知っています。

凝集度です。

となると`components`の凝集度はどれに当たるでしょう?
おそらくは最低最悪の偶発的凝集です。

凝集度については説明は省きます（一人大LTになってしまう）

技術駆動パッケージングの改善

技術駆動パッケージングの問題点が、
ある機能の低凝集と、それぞれのディレクトリ内部の偶発的凝集だと分かったので
これらを機能的凝集へと改善しましょう

技術駆動パッケージングの改善

段階的にやるのは面倒なので結果をドン!

```
src/
└ features/
  ├ login/
    ├ use-login.ts
    ├ login-page.tsx
    └ login-form.tsx
  ├ ui/
    ├ button.tsx
    ├ card.tsx
    └ ...
  └ ...
...
```

技術駆動パッケージングの改善

やったことは二つ。

- 偶発的凝集を起こしていたパッケージ、components・hooks・pagesを削除
- その中身を機能ごとにパッケージング

```
src/
└ features/
  ├ login/
    ├ use-login.ts
    ├ login-page.tsx
    └ login-form.tsx
  ├ ui/
    ├ button.tsx
    ├ card.tsx
    └ ...
  └ ...
...
```

Re:

Feature driven folder structure は何を解決する？

最後に

「銀の弾などない」（戒め）

fin

ご清聴ありがとうございました！

参考記事

Screaming Architecture - Evolution of a React folder structure

Screaming Architecture