ファイル構成

お勧めの React プロジェクトの構成は？
React はファイルをどのようにフォルダ分けするかについての意見を持っていません。とはいえ、あなたが検討したいかもしれないエコシステム内でよく用いられる共通の方法があります。

機能ないしルート別にグループ化する
プロジェクトを構成する一般的な方法の 1 つは、CSS や JS やテストをまとめて機能ないしルート別のフォルダにグループ化するというものです。

```
common/
  Avatar.js
  Avatar.css
  APIUtils.js
  APIUtils.test.js
feed/
  index.js
  Feed.js
  Feed.css
  FeedStory.js
  FeedStory.test.js
  FeedAPI.js
profile/
  index.js
  Profile.js
  ProfileHeader.js
  ProfileHeader.css
  ProfileAPI.js
```

ここでの「機能」の定義は普遍的なものではないので、粒度の選択はあなた次第です。トップレベルのフォルダの名前が思いつかない場合は、ユーザに「この製品の主な構成部品は何か」と聞いてみて、ユーザの思考モデルを青写真として使いましょう。

ファイルタイプ別にグループ化する
プロジェクトを構築する別の人気の方法は、例えば以下のようにして類似ファイルをグループ分けするというものです。

```
api/
  APIUtils.js
  APIUtils.test.js
  ProfileAPI.js
  UserAPI.js
components/
  Avatar.js
  Avatar.css
  Feed.js
  Feed.css
  FeedStory.js
  FeedStory.test.js
  Profile.js
  ProfileHeader.js
  ProfileHeader.css
```

人によってはこの方法をさらに推し進め、コンポーネントをアプリケーション内の役割に応じてフォルダ分けすることを好みます。例として、Atomic Design はこのような原則の下に作られたデザインの方法論です。ただこのような方法論は、従わなければならない厳格なルールとして扱うよりも、役に立つ見本として扱う方が多くの場合生産的であるということを忘れないでください。

ネストのしすぎを避ける
深くネストされた JavaScript プロジェクトには様々な痛みを伴います。相対パスを使ったインポートが面倒になりますし、ファイルが移動したときにそれらを更新するのも大変です。よほど強い理由があって深いファルダ構造を使う場合を除き、1 つのプロジェクト内では 3 段か 4 段程度のフォルダ階層に留めることを考慮してください。もちろんこれはお勧めにすぎず、あなたのプロジェクトには当てはまらないかもしれません。

考えすぎない
まだプロジェクトを始めたばかりなら、ファイル構成を決めるのに 5 分以上かけないようにしましょう。上述の方法の 1 つを選ぶか、自分自身の方法を考えて、コードを書き始めましょう！ おそらく実際のコードをいくらか書けば、なんにせよ考え直したくなる可能性が高いでしょう。
もしも完全に詰まった場合は、すべて 1 フォルダに入れるところから始めましょう。そのうち十分に数が増えれば、いくつかのファイルを分離したくなってくるでしょう。そのころには、どのファイルを一緒に編集している頻度が高いのか、十分わかるようになっているでしょう。一般的には、よく一緒に変更するファイルを近くに置いておくのは良いアイディアです。この原則は、「コロケーション」と呼ばれます。
プロジェクトが大きくなるにつれ、実際にはしばしば上記両方の方法が組み合わされて使用されます。ですので、「正しい」方法を最初から選択することはさほど重要ではありません。

このページを編集する