Typewind

Tailwindにさらなる追い風を

自己紹介

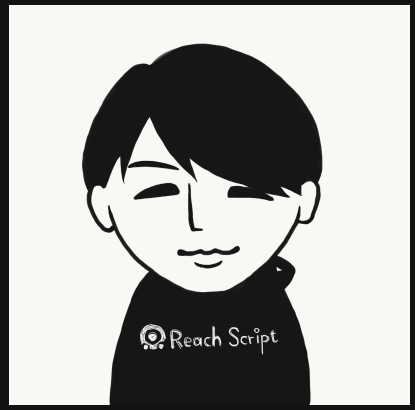
近藤 泰治

所属: Reach Script

肩書: ロジックリードエンジニア

得意分野: フロントエンド

好きなこと: 筋トレ、HHKB



Tailwind CSS 使ってますか？

Tailwind CSSとは

ユーティリティファーストのCSSフレームワーク

簡単な使い方

通常のCSS

```
<button class="button">Click me</button>
```

```
.button {
    width: 40px;
    height: 20px;
    background-color: #ef4444;
    border-radius: 0.25rem; /* 4px */
}
```

Tailwind

```
<button class="w-10 h-5 bg-red-500 rounded">Click me</button>
```

メリット

- クラス名を考える手間が省ける
- CSSファイルを作らなくて良い
- CSS設計を考えなくて良い
- Reactとの相性が良い

デメリット

- Tailwindのクラス名を覚える必要がある
- class名の記述が長くなりがちで可読性が落ちる

```
const Button = () => {
  return (
    <button className="bg-blue-500 text-white text-lg font-bold py-2 px-4 rounded shadow-sm hover:bg-blue-700">
      Click me
    </button>
  );
};
```

TypeScript 使ってますか？

TypeScriptとは

JavaScriptのスーパーセットである静的型付け言語

簡単な使い方

通常のJavaScript

```
const addNumbers = (a, b) => {
    return a + b;
}

// この処理はエラーにならずに結果として文字列の`57`が出力される 😇
const result = addNumbers(5, "7")
```

↓

TypeScript

```
const addNumbers = (a: number, b: number) => {
    return a + b;
}

// Compile Error!! 🧐
const result = addNumbers(5, "7")
```

型があれば

- コンパイル時にエラーに気づける
- 代入ミスが減る
- 補完が出る
- タイポが減る

総じて開発スピードUPにつながる☺️

TypeScript のエッセンスを Tailwind に持ち込めればハッピーになれそう

それが

Typewind

Typewindとは

Type SafeなTailwind

Typewindを使うと何が嬉しいか

- Tailwindのデメリットで挙がった「Tailwindのクラス名を覚える必要がある」を（少し）解決してくれる
- 型のおかげでclass名の補完が出るのでタイポしなくなる
- IDEのTailwindプラグインに頼らなくて良くなる

```
// 🤔
const TailWindButton = () => {
  return (
    <button className="bg-blue-500 text-white text-lg font-bold py-2 px-4 rounded shadow-sm hover:bg-blue-700">
      Click me
    </button>
  );
};

// 😎
import { tw } from "typewind";

const TypeWindButton = () => {
  return (
    <button className={tw.bg_blue_500.text_white.text_lg.font_bold.py_2.px_4.rounded.shadow_sm.hover(bg_blue_700)}>
      Click me
    </button>
  );
};
```

使い方をざっと解説

Normal Usage

- Typewindから提供される`tw`オブジェクトにアクセス、チェーンすることでTailwindのクラスを呼び出せる
- `tw`は`tailwind.config.js`の設定を元にしているので独自で定義したカラーパレット等も呼び出し可能
- Typewindのクラス名はTailwindのクラスのハイフン(`-`)をアンダースコア(`_`)に変えたもの
- `-mt-1`のような負の値(ハイフン始まり)のクラスは、`tw._mt_1`のように指定する
- `inset-1/2`のような/が含まれるクラスは、`tw.inset_["1/2"]`のように指定する
- 色の不透明度は、`tw.bg_blue_500$["25"]`のように指定する

```
const Normal = () => {
  return (
    <div className={tw._mt_1}>
        <div className={tw.inset_["1/2"]}>
            <button className={tw.bg_blue_500.text_white.text_lg.font_bold.py_2.px_4.rounded.shadow_sm.hover(bg_blue_70
              Click me
            </button>
        </div>
    </div>
  );
};
```

Modifiers

- `hover`、`before`等の疑似要素、擬似クラス
- `tw.hover()`として引数に当てたいクラスを渡す

```
const Normal = () => {
  return (
    <button
      className={tw.bg_blue_500.hover(tw.bg_blue_600).first_letter(tw.text_red_500.font_bold)}
    >
      Click Me
    </button>
  );
};
```

Important

- `important`をつけたい場合は`tw.important()`として引数に当てたいクラスを渡す

```
const Normal = () => {
  return (
    <button
      className={tw.important(tw.text_red_500).hover(tw.important(tw.text_red_600))}
    >
      Click Me
    </button>
  );
};
```

Arbitrary Values

- `text-[20px]`のようなTailwindで定義されていない値を使いたい場合の記述
- `tw.text_[20px]`のように書ける

```
const Normal = () => {
  return (
    <button className={tw.text_['20px'].py_3.px_4.bg_blue_500}>Click Me</button>
  );
};
```

他にも

- `[&:nth-child(3)]:underline`のように指定できる`Arbitrary Variants`
- 任意の親要素の横幅を基準としてスタイルを当てることができる`Container Query`

などにも対応

まとめ

- IDEのTailwindプラグインよりも補完が確実で早い（気がする）のでストレスが減る
- かつ、プラグインではWarningだがTypewindならErrorになってくれる
- セットアップの手軽さの割にメリットが多い
- TypeScript最高

参考文献

- https://typewind.dev/
- https://github.com/mokshit06/typewind
- https://zenn.dev/taiji/articles/7cae4a672668d3

END