

shadcn/uiで考えるコンポーネント設計

ゆめみ×LayerX×サイボウズ3社合同フロントエンドカンファレンス北海道2024後夜祭@東京 2024/09/06



About

- Infixer(Ryo Katsuse)
- 株式会社ゆめみ
- フロントエンドエンジニア
- 放送大学3年生
- アウトプット
 - X (Twitter)
 - Consense (旧scrapbox)
 - ブログ
 - GitHub
- 北海道で食べた美味しかったもの
 - ジンギスカン、〆パフェ



ちょっと宣伝

アクセシビリティLT会 #2をLINEヤフーさまと共催で大阪にて行います！



開催：2024/10/10 (木):LINEヤフー株式会社 大阪グランフロントオフィス（オンラインでも配信！）

YUMEMI.growのCompassページにて近日公開！

前回のアクセシビリティLT会の内容

お品書き

- shadcn/uiとは
 - コンポーネントの思想
- cvaを用いた設計
- formコンポーネントについて
- 案件でつかったみた感想
- まとめ

本日は、採択されなかったプロポーザル（LT5分枠）についてお話しします！

shadcn/uiの話をしますが、最近はReact Ariaにハマっています。。。！



fortee

shadcn/uiから考える
コンポーネント設計

by Ryo Katsuse / @ryo_kts



shadcn/uiとは

Build your component library

Beautifully designed components that you can copy and paste into your apps.

Get Started GitHub

Mail

Dashboard

Cards

Tasks

Playground

Forms

Music

Auth

Alicia Koch

Inbox

All mail

Unread

Inbox 128

Drafts 9

Sent

Junk 23

Trash

Archive

Social 972

Updates 342

Forums 128

Shopping 8

Promotions 21

William Smith

11 months ago
Meeting Tomorrow

Hi, let's have a meeting tomorrow to discuss the project. I've been reviewing the project details and have some ideas I'd like to share. It's crucial that we align on our next steps to ensure the project's success.

Alice Smith 11 months ago
Re: Project Update

Thank you for the project update. It looks great! I've gone through the...

work important

Bob Johnson over 1 year ago
Weekend Plans

Any plans for the weekend? I was thinking of going hiking in the nearby...

personal

Emily Davis over 1 year ago
Re: Question about Budget

I have a question about the budget for the upcoming project. It seems like...

work budget

Michael Wilson over 1 year ago
Important Announcement

I have an important announcement to...

WS William Smith
Meeting Tomorrow
Reply-To:...

Oct 22, 2023, 9:00:00 AM

Hi, let's have a meeting tomorrow to discuss the project. I've been reviewing the project details and have some ideas I'd like to share. It's crucial that we align on our next steps to ensure the project's success.

Please come prepared with any questions or insights you may have. Looking forward to our meeting!

Best regards, William

Reply William Smith...

Send

shadcn/uiとは

- インストール不要で、ソースコードをコピペだけで使用できるコンポーネント集
- コンポーネントを好きなようにカスタマイズ可能
- 依存関係を気にせず好きなコンポーネントを好きな分だけ導入可能
- v0で吐き出されるコードはshadcn/uiベースのもの

v0にカンファレンスのタイムテーブル的なものを作らせてみた！

shadcn/ui公式サイト

実際にプロジェクトで導入してみた

- 小・中規模の決済システムのようなアプリケーション
- システムの構成上凝ったデザインがなくシンプルなUI
- Next.jsのPage Routerでの開発
- 開発期間が短い
 - コンポーネント開発に時間を割けない

上記を踏まえた上で、プリミティブなコンポーネントがまとめたshadcn/uiを採用✌️

- PandaCSS
 - shadcn/ui以外の個別のレイアウトなどを実装する際に採用したが、オーバーヘッドがあったり、TailwindCSSの世界線で収まるかと判断したため早い段階でやめた

shadcn/uiは、プリミティブでシンプルなコンポーネントが揃っているので、大規模で複合的なコンポーネントや、コンポーネントの数が多いようなアプリケーションでは向いていないかもしれない

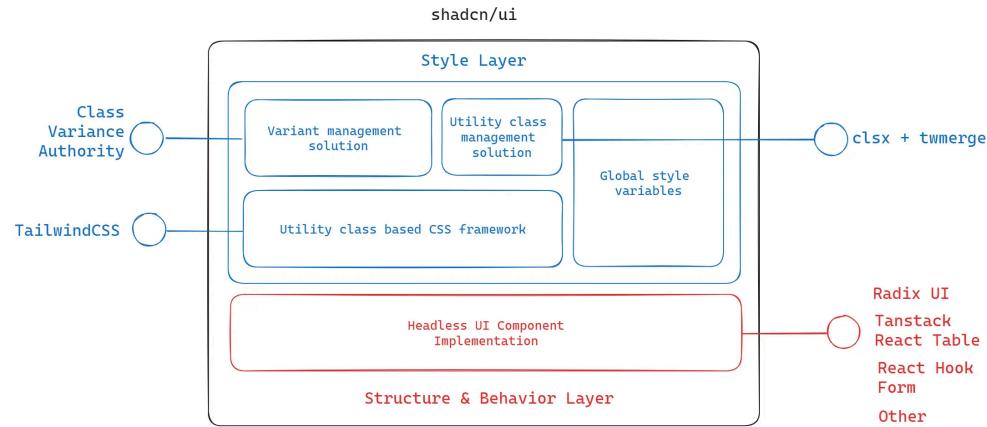
Mantineなど最初からコンポーネントが豊富なものが良さそう

shadcn/uiの思想 構造とスタイルの分離

構造

- ヘッドレスUI
 - RadixUIをベースにアクセシビリティ対応
だったりインタラクションの部分を提供
している
 - DatePickerはReact DayPickerを使っている
 - フォームについてはReact Hook Form、
テーブルについてはTanStack Tableなど

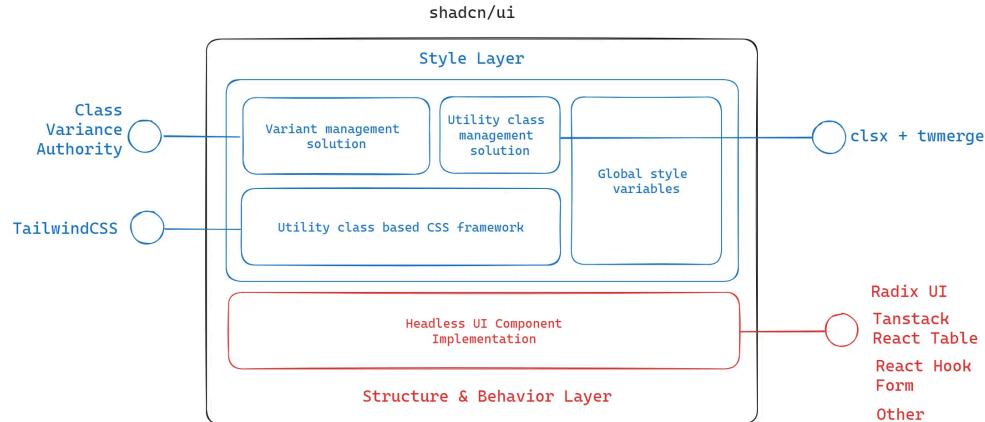
The anatomy of shadcn/ui



スタイル

- コアな部分のCSSがTailwindCSS
- class文字列の連結など、ユーティリティな管理にはtwMergeとclsx
- グローバルなスタイルはTailwind.config
- 各種のVariant管理にはCVAを使用（後述します）

The anatomy of shadcn/ui



なぜこうなっているのか？

Introductionに記述がある

Why copy/paste and not packaged as a dependency?

The idea behind this is to give you ownership and control over the code, allowing you to decide how the components are built and styled. Start with some sensible defaults, then customize the components to your needs. One of the drawbacks of packaging the components in an npm package is that the style is coupled with the implementation. The design of your components should be separate from their implementation.

コードの所有権と制御を与えます！

コンポーネントの構築方法とスタイルを決定できるようにすることが重要と考えています。

パッケージの欠点の1つはパッケージ化すると、スタイルと実装が密結合なっている。

コンポーネントのデザインは実装と切り離すべき！！

CVAについて

CVAとは

- プライマリー、セカンダリーボタンのようなコンポーネントを実装するとき
 - classnamesやclsxなどを使って条件分岐によってスタイルを切り替える
 - そもそもコンポーネントを分割してしまう
 - 大量のpropsがあると何を渡せばいいかわからなくなる。。。
- そんな悩みを解決してくれるのがCVA！
- 最終的に出力されるCSSがどのような状態のスタイルなのかが構造化されているので見やすい
- FigmaのVariantsの概念をそのままコードに落としめるようなイメージ

ちなみに、TailwindCSSの場合は、cvaより拡張性の高いTailwind Variantsがあります

こういうやつ

```
const buttonVariants = cva(  
  "inline-flex items-center justify-center whitespace nowrap rounded-md text-sm font-medium ring-offset-background transition-all",  
  {  
    variants: {  
      variant: {  
        default: "bg-primary text-primary-foreground hover:bg-primary/90",  
        secondary: "bg-secondary text-secondary-foreground hover:bg-secondary/80",  
      },  
      size: {  
        default: "h-10 px-4 py-2",  
        sm: "h-9 rounded-md px-3",  
        lg: "h-11 rounded-md px-8",  
        icon: "h-10 w-10",  
      },  
    },  
    defaultVariants: {  
      variant: "default",  
      size: "default",  
    },  
  },  
)
```

Buttonコンポーネントの構造

```
export interface ButtonProps
  extends React.ButtonHTMLAttributes<HTMLButtonElement>,
  VariantProps<typeof buttonVariants> {
  asChild?: boolean
}

const Button = React.forwardRef<HTMLButtonElement, ButtonProps>(
  ({ className, variant, size, asChild = false, ...props }, ref) => {
  const Comp = asChild ? Slot : "button"
  return (
    <Comp
      className={cn(buttonVariants({ variant, size, className }))}
      ref={ref}
      {...props}
    />
  )
}
)
Button.displayName = "Button"

export { Button, buttonVariants }
```

Dialogコンポーネント

- それぞれのパートを組み合わせて使うようになっている

```
<Dialog>
  <DialogTrigger asChild>
    <Button variant="primary">open</Button>
  </DialogTrigger>
  <DialogContent>ヘッダー</DialogContent>
  <DialogContent className="flex flex-col">
    <p>コンテンツ</p>
  </DialogContent>
  <DialogFooter>フッター</DialogFooter>
</Dialog>

// 使用例
<Dialog
  triggerButton={<Button variant="primary" onClick={dialogOpen}>open</Button>}
  header='ヘッダー'
  content={<p>コンテンツ</p>}
  footer='フッター'
/>
```

Formコンポーネント

Formコンポーネント

- shadcn/uiにはFormコンポーネントがある
- 内部的にReact Hook Formを使っている
- 実際のプロジェクトでもFormコンポーネントを使った。
- 普段からReact Hook Formを使っていればそこまでハマることはない。

```
<Form {...form}>
  <form className="w-60 px-2" onSubmit={handleSubmit(onSubmit)}>
    <FormField
      control={form.control}
      name="label"
      render={({ field: { value, onChange } }) => (
        <FormItem>
          <FormLabel>ラベル</FormLabel>
          <FormControl>
            <NumberInput
              value={value}
              onChange={(value) => {
                onChange(value.value);
              }}
              isError={isDefined(form.formState.errors.label)}
            />
          </FormControl>
          <FormMessage />
        </FormItem>
      )}
    />
    <p>Current value is: {form.watch('label')}</p>
    <Button type="submit" className="mt-4">
      Submit
    </Button>
  </form>
</Form>
```

しかしReact Hook Formは使いたくない！

- React Hook Formは、ドキュメントが分かりにくい！！！（個人の主観）
- React Server Componentの登場によって雲行きが怪しくなっている。
- なんかいいライブラリがないかなー
- Conformといライブラリがあるみたいだよ
- シンプルだし、機能的にも良さそう！
- *shadcn/ui*はプリミティブな*Input*などが揃っているので、*Form*コンポーネントだけネイティブの*form*に置き換えれば実装できる！！！！

案件でつかったみた感想

案件でつかったみた感想

- 実装するアプリケーションが凝ったデザインがシンプルで、コンポーネントの数も多くない場合は選択肢の一つ
- 実際にほとんどコピペだけの状態で構造は利用して、CSSはデザイントークン（色、タイポグラフィなど）の微修正だけでコンポーネントがすぐにできた
- フォームコンポーネントについては、特にハマることもなかった。
- shadcn/uiではない独自コンポーネントでもcvaを使ってCSSを管理することで統一感が生まれた
- classNameを渡すことができてしまうので、ルールなどをチームと話し合うと良さそう

```
const BlockVariants = cva('flex w-full items-center justify  
variants: {  
  bgColor: {  
    primary: 'bg-main-primary-dark',  
    error: 'bg-unique-error',  
    disabled: 'bg-text-disable',  
  },  
},  
});  
  
const Block: FC<BlockProps> = ({ variant, amount }) => {  
  return (  
    <div className={BlockVariants({ bgColor: variant })}>  
      <p className="text-large text-base-white">{TypeText[v  
        <div className="text--block text-base-white">  
          <div className="inline-block">  
            <span>{numericFormatter(amount.toString(), { thou  
              <span>円</span>  
            </div>  
            <p className="inline-block px-2 align-bottom text-si  
          </div>  
        </div>  
      );  
  
      export { Block };
```

まとめ

- shadcn/uiはプリミティブなコンポーネントが集まったコンポーネントを好きな分だけ使える便利なもの
- 構造とスタイルのレイヤーに分離されていて、カスタマイズが自由にできるので、柔軟に開発ができる！
- Variantはいいぞ！
- Formコンポーネントは使わなくても大丈夫！