

でざトーーク！

15分でわかるVue.js
&
15分でわかるVueプロジェクトの構造



Vue.jsとは？

- JavaScriptのフレームワーク

- フレームワーク

…アプリケーション開発を簡単にしてくれる
骨組みのようなもの

- 小規模でも大規模でも使えると評判



どうしてVueは人気なのか？

- 学習コストが低い、取っつきやすい
- DOMの更新を自動化 「データバインディング」
- 保守性、再利用性を高める 「コンポーネント」

などいろいろ

Vueの基本的な仕組み

すべてはVueインスタンスから始まる

Vueインスタンスを生成、
「ここからここまでの範囲でVueを使うよ」
と定義（id=appのdiv内を指定するのが慣例）

#appっていうel(要素)の中でVue使えるよ～

```
<div id='app'>  
</div>
```

index.html

```
new Vue({  
  el: '#app'  
})
```

main.js

Vueインスタンスはdataを持てる

UIに関係するデータは、
インスタンスのdataに登録（jsonっぽい書き方）

```
data() {  
  return {  
    msg: "こんにちは！",  
    fruits: ["りんご", "パイナップル", "いちご", "みかん"],  
    area: 'none'  
  }  
}
```

数値、文字列、配列、真偽値など

コードが散らからない&分業しやすい書き方

```
<template>
  <div class="example">
    <span class="title">{{ text }}</span>
  </div>
</template>
```

```
<script>
  export default {
    name: 'Example',
    data() {
      return {
        text: 'example'
      }
    }
  }
</script>
```

```
<!-- scoped CSS -->
<style scoped>
  .title {
    color: #ffbb00;
  }
</style>
```

template (htmlに似てる部分) ,
script (jsの処理) ,
CSS (scssも使えます) を
1つのファイルに書ける

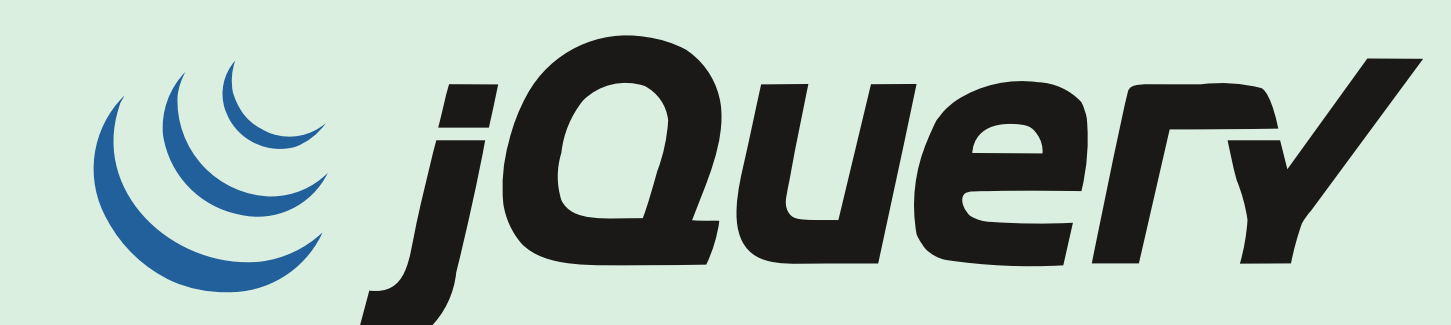
すごいぞデータバインディング

すごいぞデータバインディング ①

- ・ マス タ ッ シ ュ

データに変更があったら自動で表示を更新

いちいちDOM更新の処理を書く必要がなくて楽！



```
<div class="hello">  
  <h1>{{ msg }}</h1>  
  <h2>Essential Links</h2>  
</div>
```

```
$(".cand-answer").each(function(index){  
  $(this).text(ans_cands[index]);  
});
```

すごいぞデータバインディング ②

- ディレクティブ（v-から始まる便利なもの）

v-for …繰り返しの描画、リストなどでよく使われる

v-if …条件によって表示を変えられる

```
<?php foreach ($prefs as $key => $val) : ?>
<option value="<?php echo $key; ?>"><?php echo $val; ?></option>
```

php

```
<select v-model="univ">
  <option v-for="univ in univList">{{ univ.name }}</option>
</select>
```

V

```
<?php if ($age < 20) : ?>
  <p>未成年です.</p>
```

php

```
<div v-if="uploadedImage">
  
</div>
```

V

すごいぞデータバインディング ②

- ディレクティブ（v-から始まる便利なもの）

v-on …指定したイベントが起きたら処理をする

v-model …フォーム入力と描画内容を同期する

面倒なDOM操作はVueがやってくれます！

```
<tr>  
  <input type="number" v-model="age" v-on:blur="ageValidate"><span> {{ age }}</span>  
</tr>
```

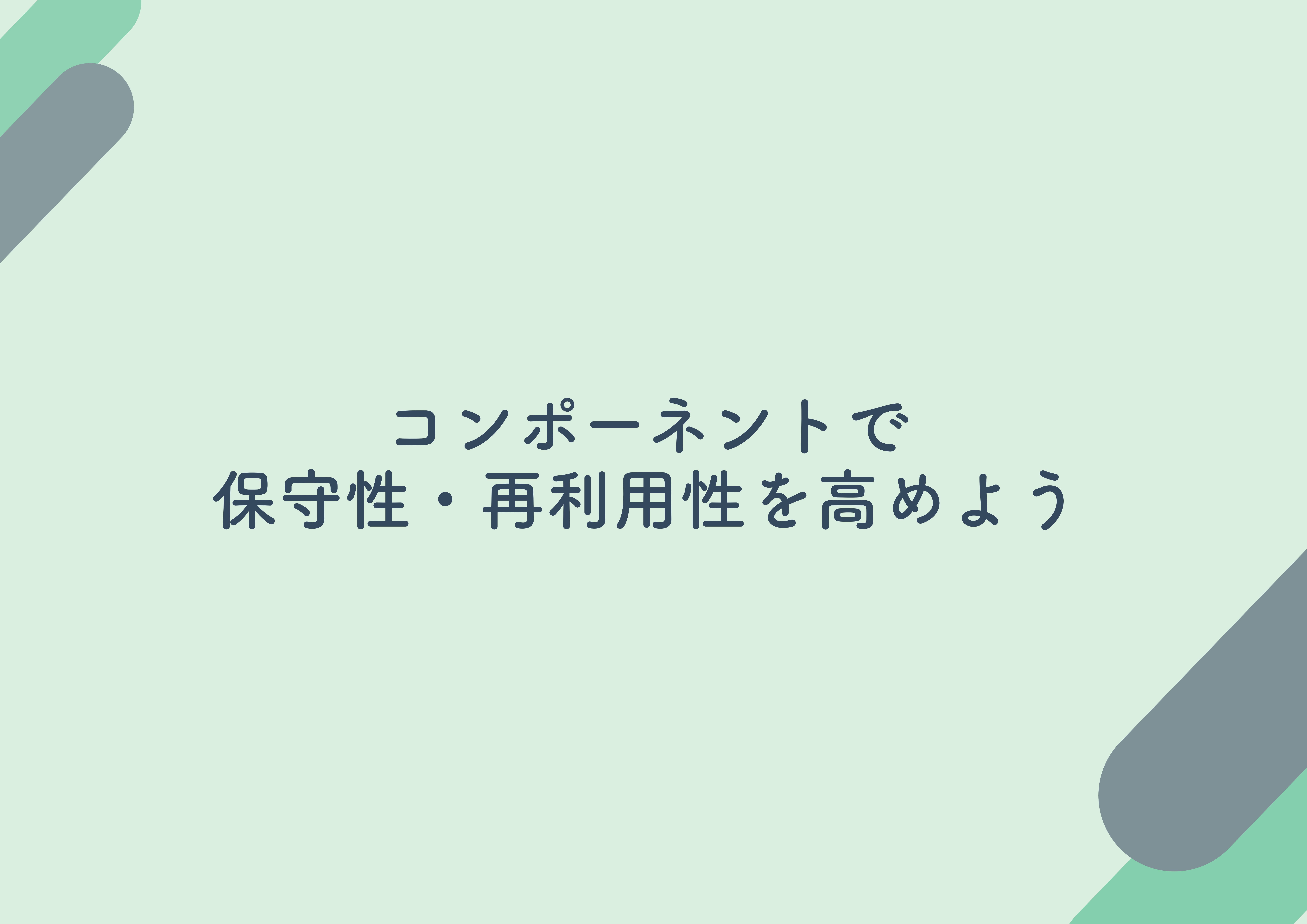

Vueで開発スピード爆上がり

- ・ホットリロード

VueCLI（後ほど説明します）で使える神機能
コードを編集・保存すると変更箇所を自動で反映
リロードいらずでサクサク作業できる

Svelte勉強会でも出てきたように、これはVue特有の機能ではないですが紹介



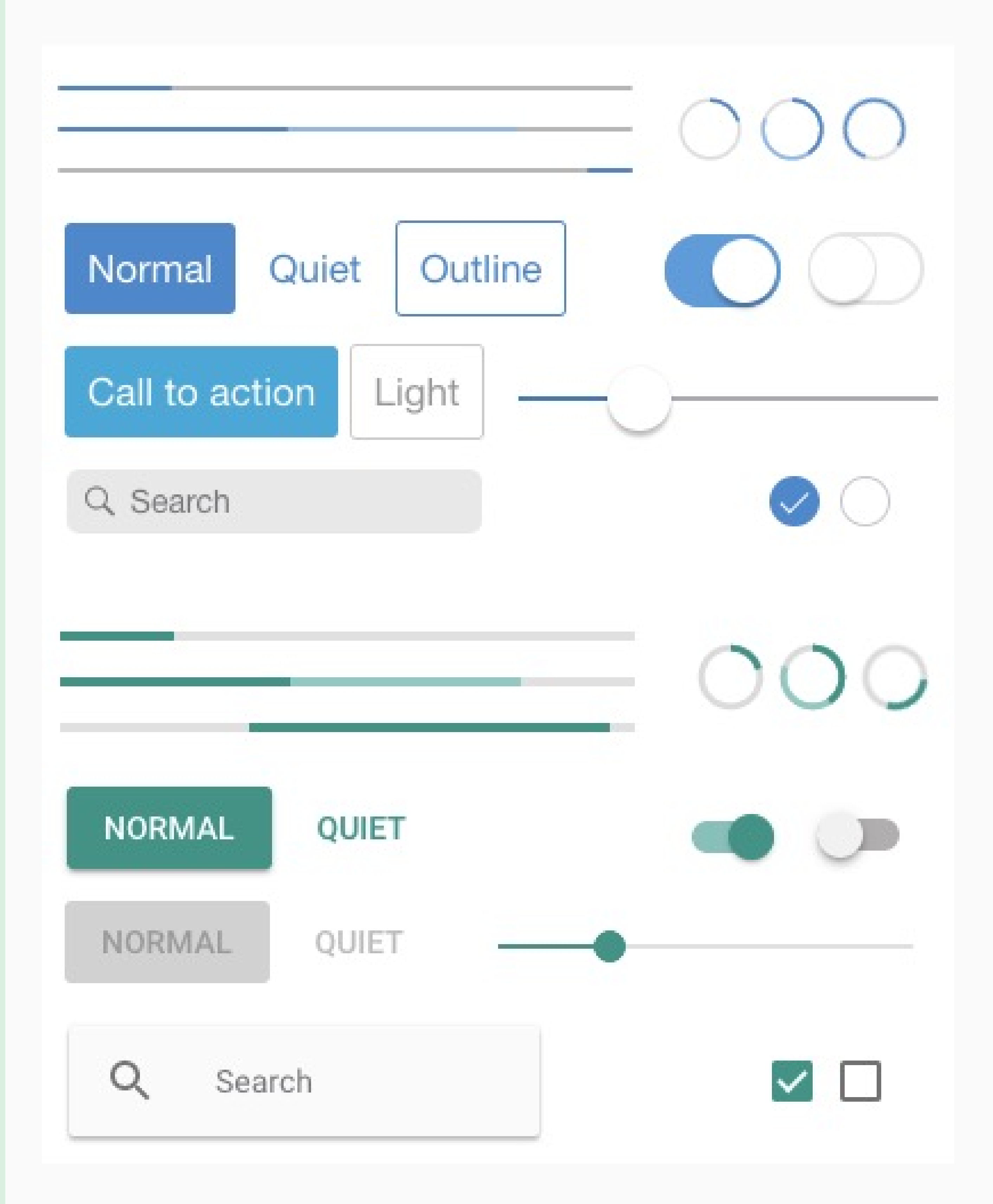
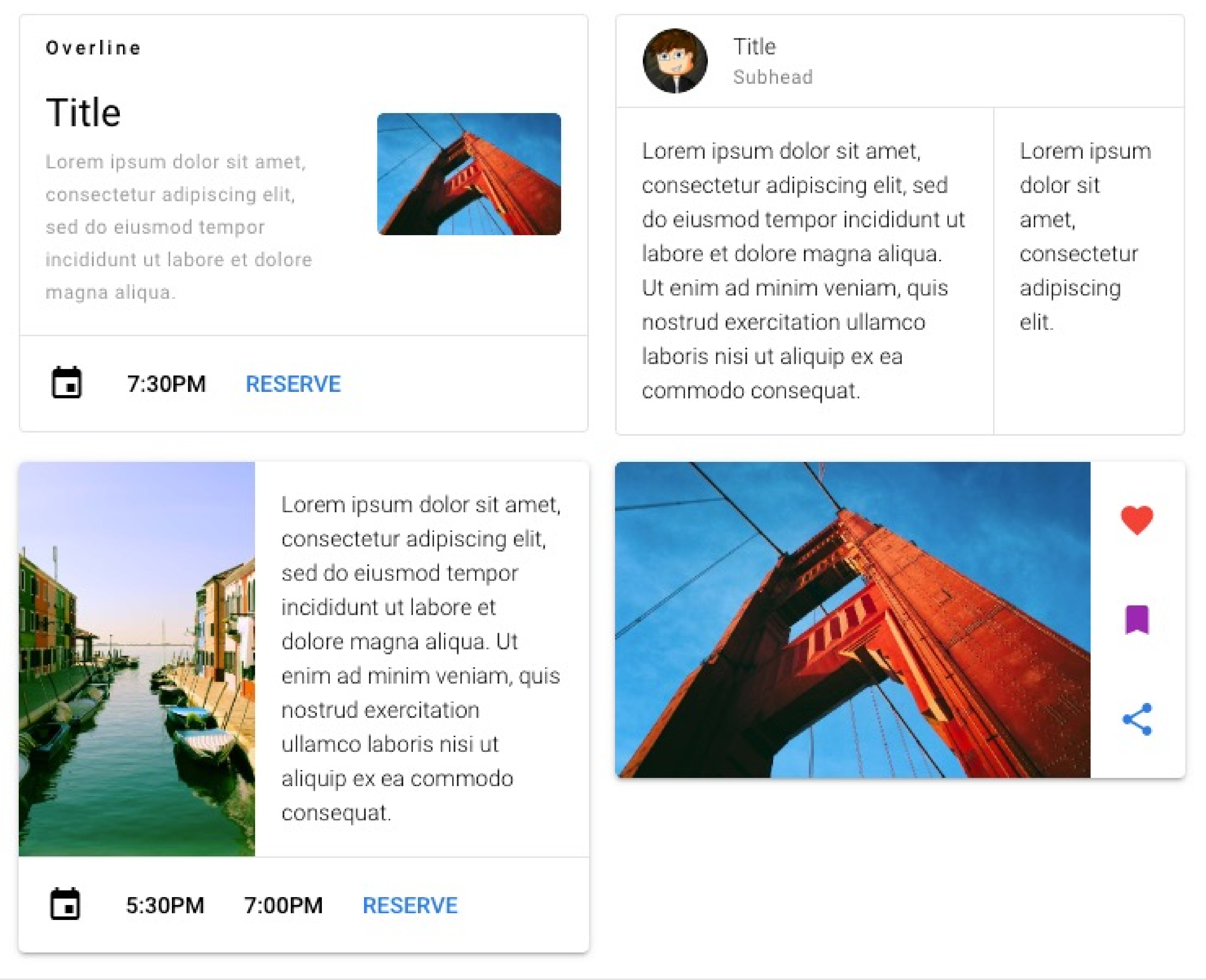


コンポーネントで
保守性・再利用性を高めよう

コンポーネントとは？

- ・イメージはSymphonyのパーシャルに近く、考え方はAtomicDesignに近い
- ・アプリのUIを機能ごとに分割して開発
- ・よく使う部品をコンポーネントにすれば、ファイルを再利用できてとても便利！

有名なコンポーネント集



コンポーネントとは？

メリット

- ・保守の工数が減る！
- ・謎ファイル、謎コードの発生を防げる
 - …コンポーネントごとにjs, cssを書くため
どこで使われているか不明なコードが
生まれにくくなる

コードが散らからない&分業しやすい書き方

```
<template>
  <div class="example">
    <span class="title">{{ text }}</span>
  </div>
</template>
```

```
<script>
  export default {
    name: 'Example',
    data() {
      return {
        text: 'example'
      }
    }
  }
</script>
```

```
<!-- scoped CSS -->
<style scoped>
  .title {
    color: #ffbb00;
  }
</style>
```

template (htmlに似てる部分) ,
script (jsの処理) ,
CSS (scssも使えます) を
1つのファイルに書ける

コンポーネントとは？

デメリット

- どんな分割をするかの共通認識を作る難しさ
- 将来の変更、拡張を予測する設計の難しさ



コンポーネントの「親子関係」

コンポーネントの親子関係

- ・コンポーネントの中にコンポーネントを配置することも＝「親子」

- ・例…カルーセル

カルーセル全体を使いまわしたい
カードもコンポーネント化したい

→カードコンポーネントをカルーセルコンポーネント内に配置

