



在前端时代，保护你的软件资产

ui-model, 跨框架复用



促进软件开发领域知识与创新的传播



关注InfoQ官方信息
及时获取QCon软件开发者
大会演讲视频信息



扫码，获取限时优惠



全球架构师峰会 2017 [深圳站]

2017年7月7-8日 深圳·华侨城洲际酒店

咨询热线：010-89880682



全球软件开发大会 [上海站]

2017年10月19-21日

咨询热线：010-64738142



官方网站

- ui-model.com

官方仓库

- github.com/ui-model

Angular官网

- angular.io (英文)
- angular.cn (中文)

关于我



ThoughtWorker

Google 开发技术专家 - GDE

Angular Contributor

Angular 官方文档中文版译者

《AngularJS 深度剖析与最佳实践》作者

《Angular 权威指南》（ng-book2）译者

前端，生存大挑战

现实：疯狂的前端

- 每六个月技术翻新一次
- 以不变应万变：把握技术内核

个人：如何复用你的经验？

个人：如何复用你的代码？

组织：如何积累组织级软件资产？

组织：如何调整组织架构？

关注点分离 - SoC

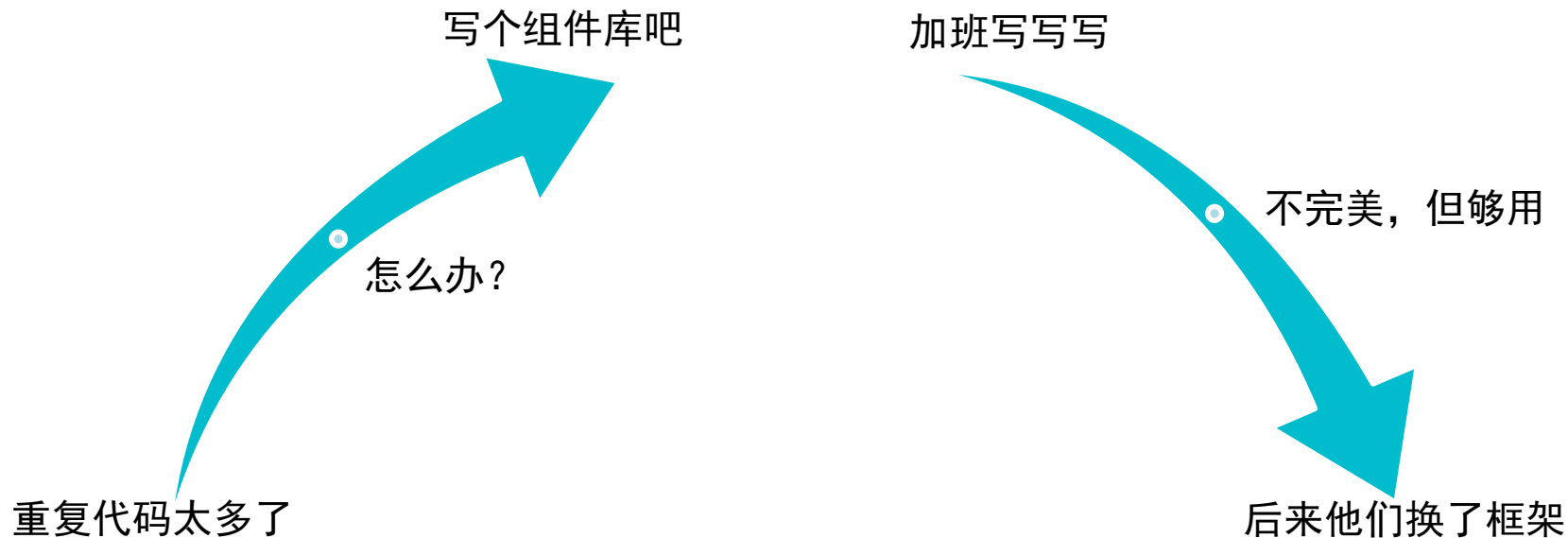
- 不要让不同的关注点混杂在一起，以免增加不必要的复杂度

单一职责原则 - SRP

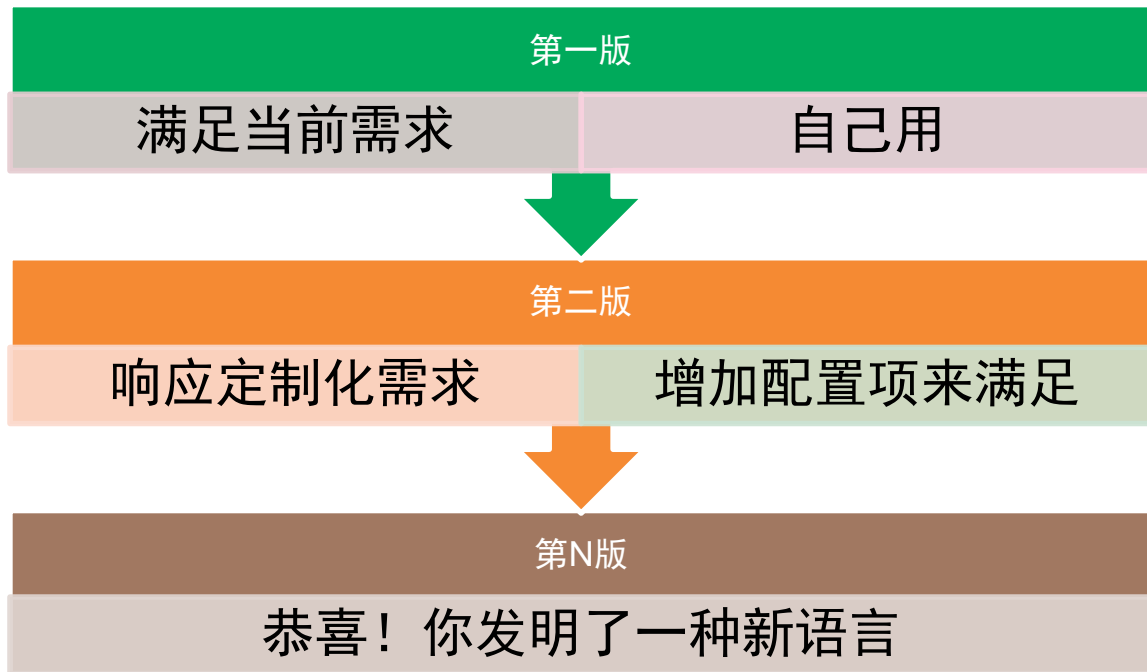
- 让各个部分具有明确且单一的职责
- 变化是BUG的原因，职责是变化的原因

组件为什么难以复用？

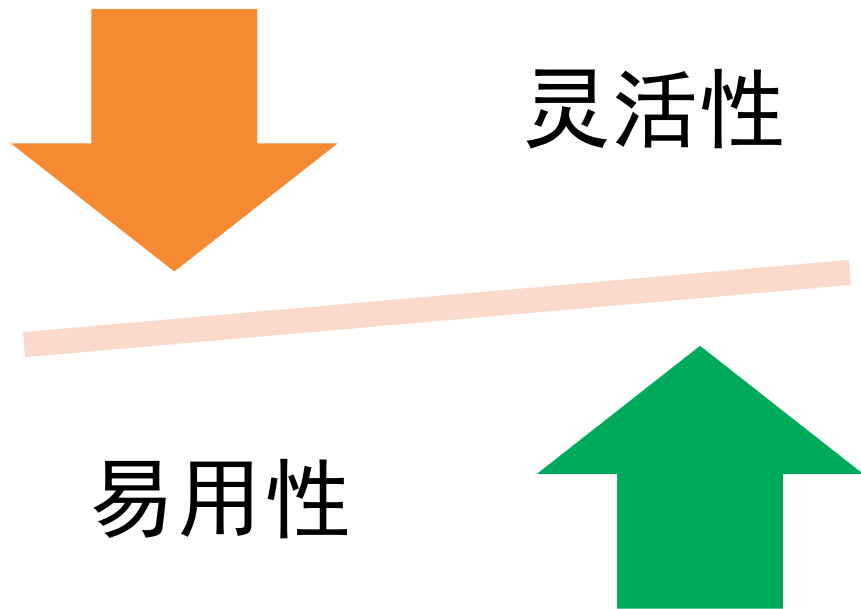
和JS / CSS框架绑得太紧



配置复杂



在定制和使用上的两难抉择



根本的原因在哪里？

组件难以复用的根本原因

组件根本不是用来复用的

- 它原本的目的是切分系统
- 本质上，JS组件要复用的是交互逻辑

组件违背了SoC原则

- 接下来我们会剖析它

什么是界面？

界面的三个关注点

内容

- 你想告诉用户什么？ - HTML

样式

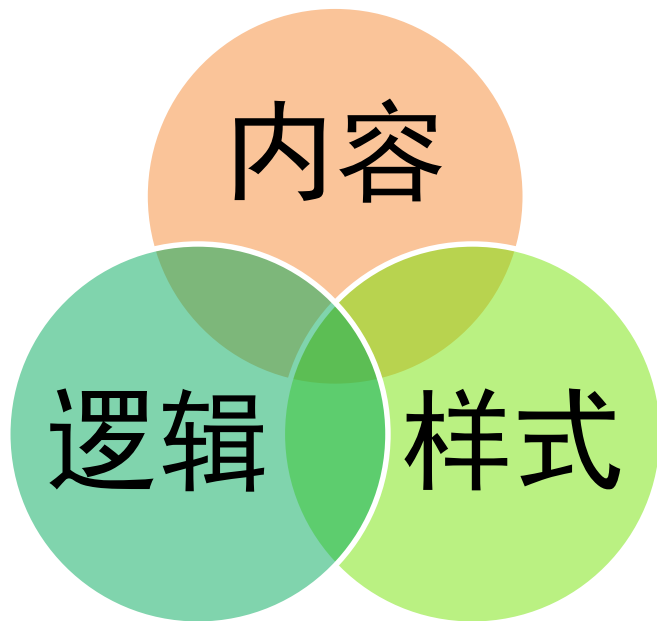
- 你想给用户传达什么情绪？ - CSS

逻辑

- 如果用户这么做，你该怎么办？ - JS

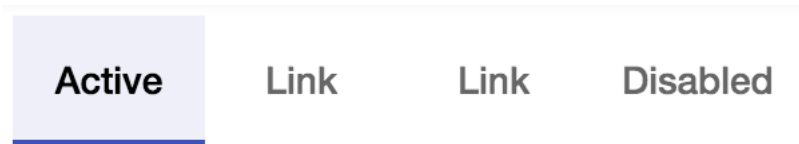
总之

组件试图同时满足这三个关注点



深入思考用户界面

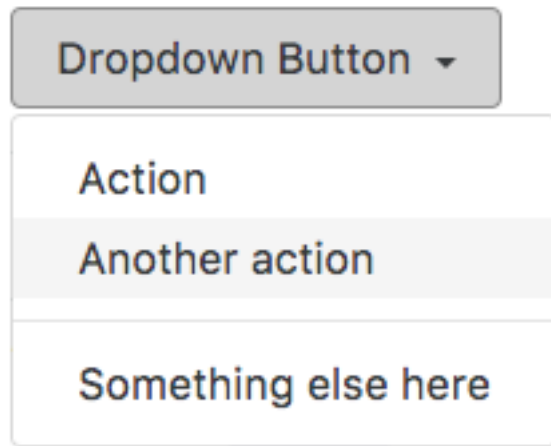
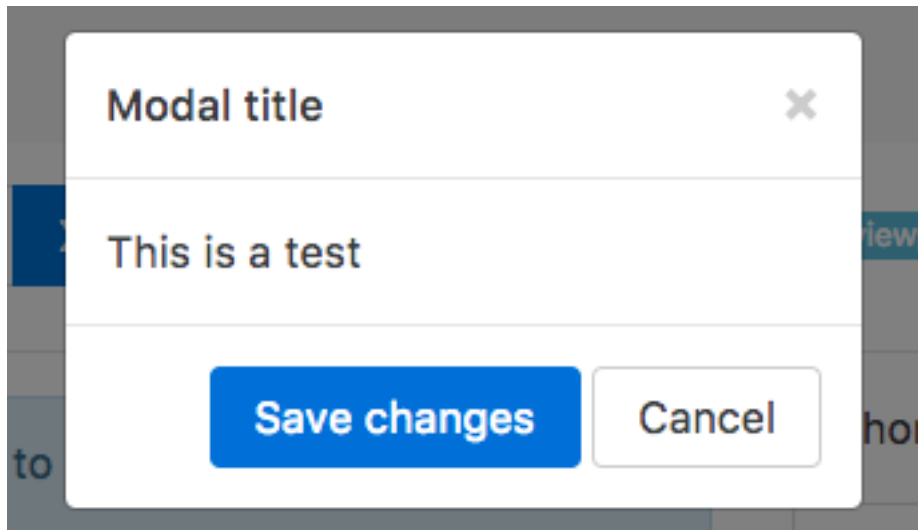
思考 - 为什么说这些组件本质上是一样的？（难度：0）



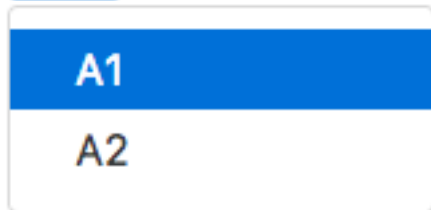
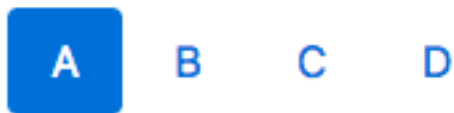
思考 - 为什么说这些组件本质上是一样的？（难度：1）



思考 - 为什么说这些组件本质上是一样的？（难度：2）



思考 - 为什么说这些组件本质上是一样的？（难度：3）



来点具体的！

```
interface Toggle {  
    isOn: boolean;  
    open(): void;  
    close(): void;  
    toggle(): void;  
    changed: Observable<boolean>;  
}
```

增加可读性

- 别名属性isOff
- 别名方法turnOn/turnOff

54行

- 实现代码

79行

- 测试代码

```
<div uiToggle #img="uiToggle">  
    
    
</div>
```


开关逻辑 - 复用

Angular 代码：

```
toggle = new Toggle();
```

```
  

```

React 代码：

```
toggle = new Toggle(this, 'toggle');
```

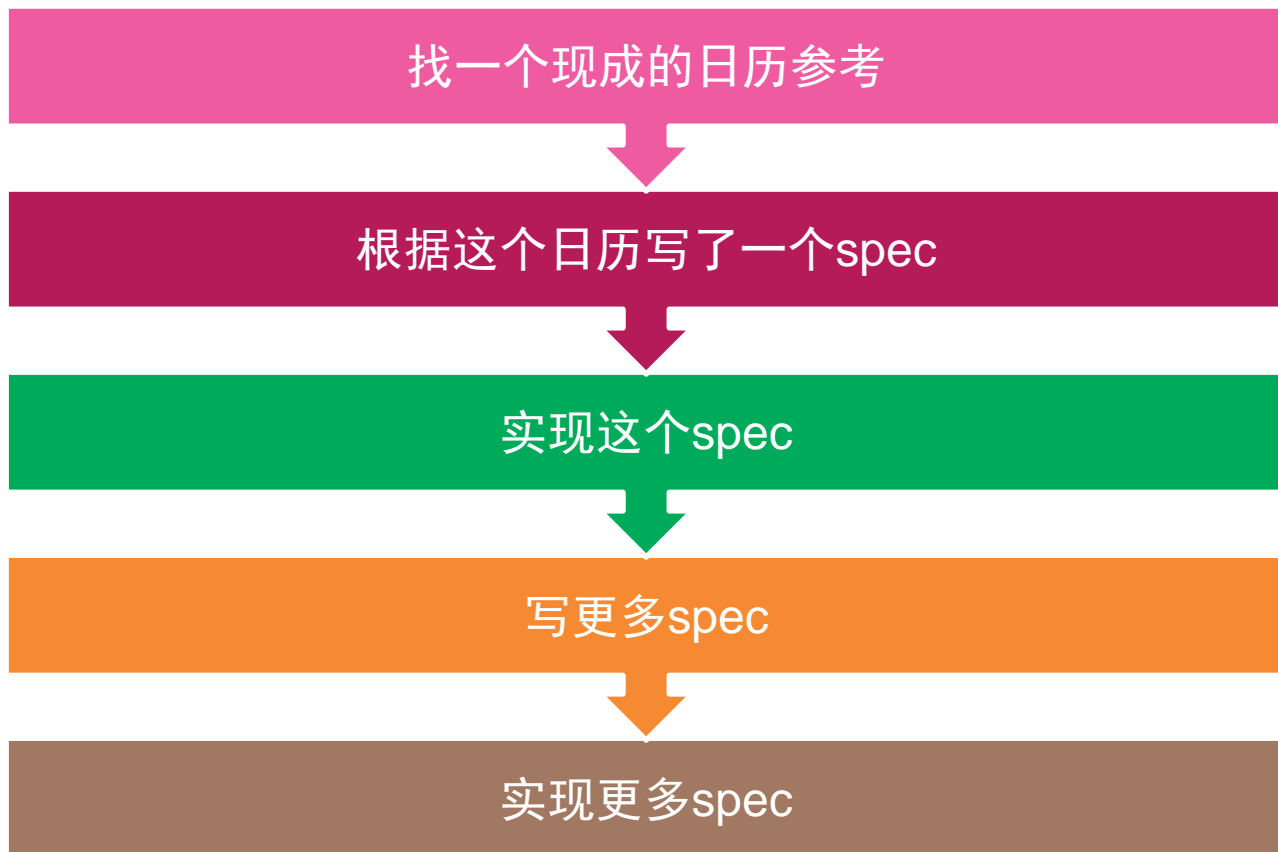
```
render() {  
  const toggle = this.toggle;  
  if (toggle.isOff) {  
    return  toggle.turnOn()}/>;  
  } else {  
    return  toggle.turnOff()}/>;  
  }  
}
```

来点复杂的！

日历模型 - 接口

```
interface Calendar {  
    value: Date;  
    year: number;  
    month: number;  
    weeks: number[];  
    getDays(week: number): Date[];  
    goTo(date: Date): void;  
    goToPrevMonth(): void;  
    goToNextMonth(): void;  
    isToday(date: Date): void;  
    isActive(date: Date): void;  
    isWeekEnd(date: Date): void;  
    readonly weekdayNames: string[];  
    readonly monthNames: string[];  
    ...  
}
```

日历模型 - 实现



Angular的用法

- 创建Calendar对象
- 把属性和方法绑定到模板中
- 代码参见<https://github.com/ui-model>

React的用法

- 创建Calendar对象
- 把this作为StateListener传给构造函数
- 在jsx中使用calendar的属性和方法

日历模型 - React代码骨架

```
render() {
  return <table>
    <caption>
      <button type="button" tabIndex="-1" className="btn btn-sm btn-secondary float-xs-
left" onClick={() => this.calendar.goToPrevMonth()}
        title="Previous Month"> &lt; </button>
      {this.calendar.value ? this.calendar.value.toLocaleDateString() : '- Choose -'}
      <button type="button" tabIndex="-1" className="btn btn-sm btn-danger float-xs-
right" onClick={() => this.calendar.clear()}>X</button>
      <button type="button" tabIndex="-1" className="btn btn-sm btn-secondary float-xs-
right" onClick={() => this.calendar.goToNextMonth()} title="Next Month"> &gt; </button>
    </caption>
    <thead>
      <tr> <th /> {this.columns} </tr>
    </thead>
    <tbody> {this.weeks} </tbody>
  </table>
}
```

ui-model实现的只是交互逻辑

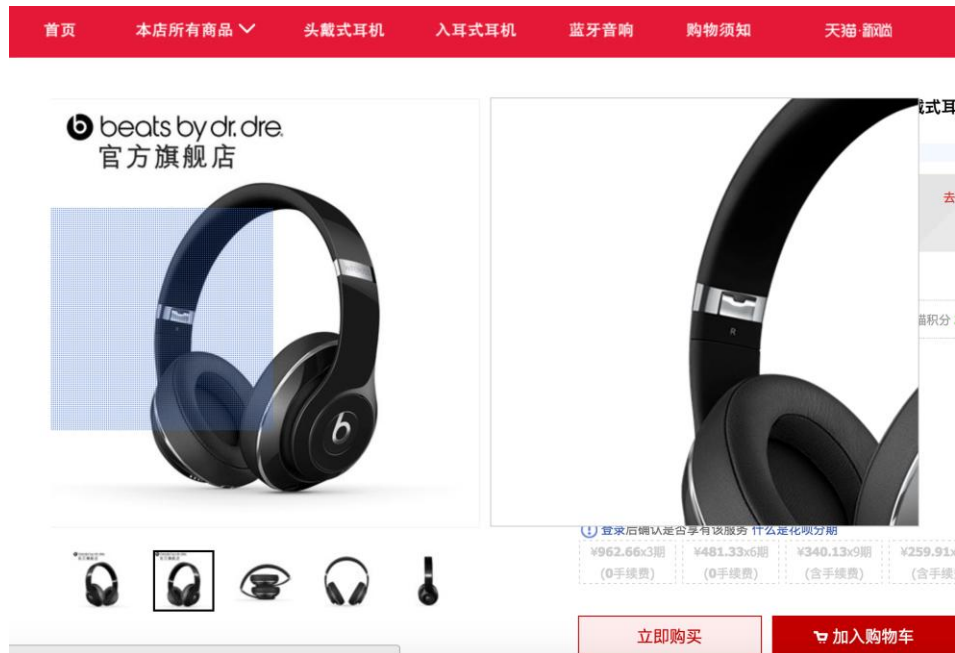
- 纯模型，不涉及界面外观和信息
- 用tdd全面覆盖

把“组件”只当做范例实现

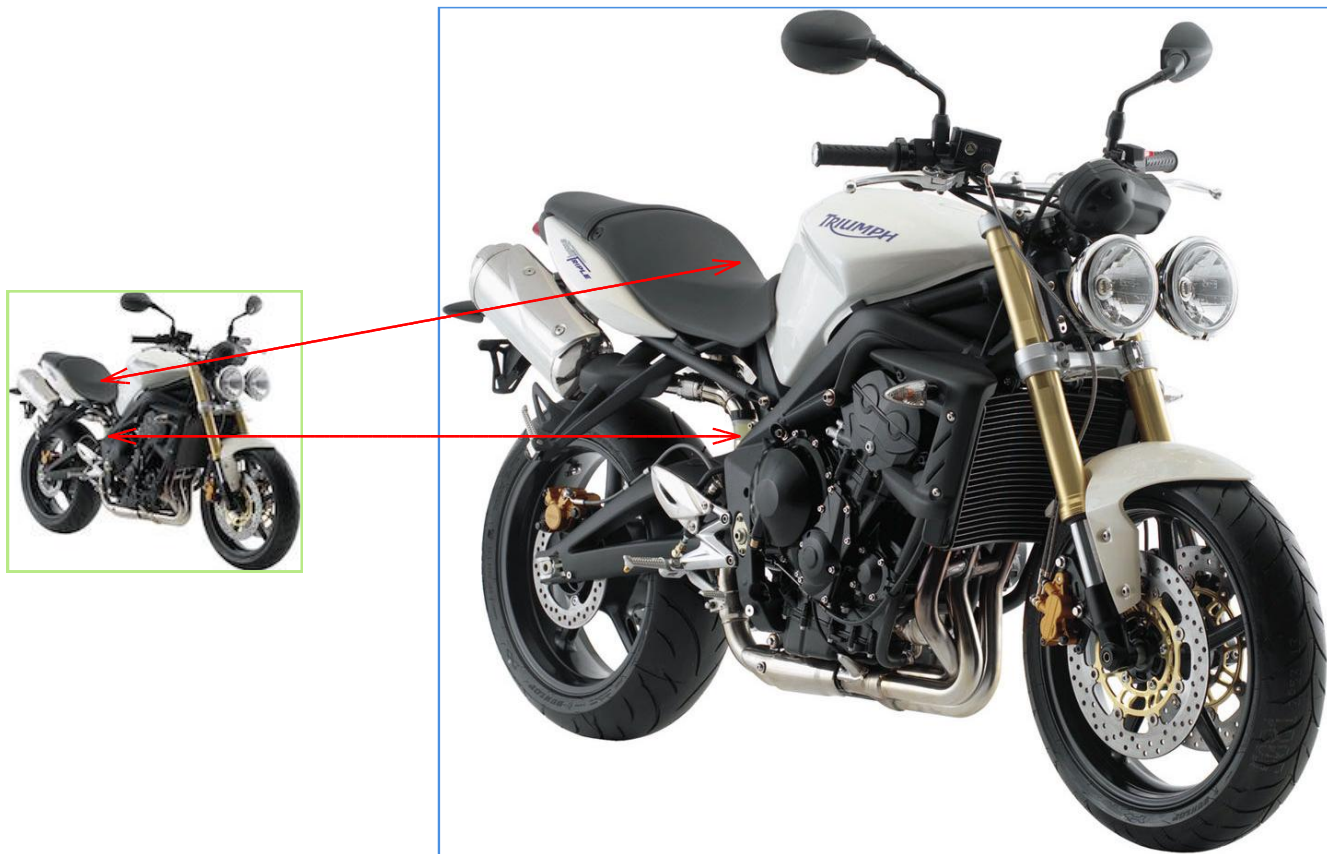
- 合适就用，不合适就Copy并修改
- 组件不涉及核心逻辑，不容易出错

来点更复杂的！

如何实现



放大镜 - 答案



Viewport

计数器

无尽滚动

分页器

超高菜单

轮播

153,9⁴⁴



ui-model 只能用来实现交互逻辑吗？

商品管理

- 添加
- 移除
- 增减
- 列表

价格预览

- 单价
- 总价
- 优惠

是否似曾相识？

ui-model 是一种思想

尽可能SoC

- 越纯粹越容易复用

简化依赖，减少耦合

- 写框架无关的代码，不要让自己被锁死

基于HTML/CSS进行定制

- HTML/CSS本来就是用来实现定制的基础设施

如何抽取 ui-model ?

做减法！

- 内容不应该出现在接口中
- 样式不应该出现在接口中

设计用法

- 先写客户代码来试验（可读性优先）
- 再用TDD方式去实现

提高友好性

- 添加派生属性
- 添加属性、方法的别名

架构中的ui-model

UI

JS
Frameworks

ui-model

CSS
Frameworks

Angular

React

...

Toggle

Select

Multi-
Select

...

Bootstrap

Semantic

...

ui-model的概念与目前的实现

概念层

- 提高抽象层次，专注交互逻辑
- 贯彻SoC和SRP
- 框架中立

实现层

- github.com上的ui-model组
- 核心库 + 框架绑定库 + Showcase
- 目前已经做了核心库和Angular绑定/Showcase以及一个日历模型的React Showcase

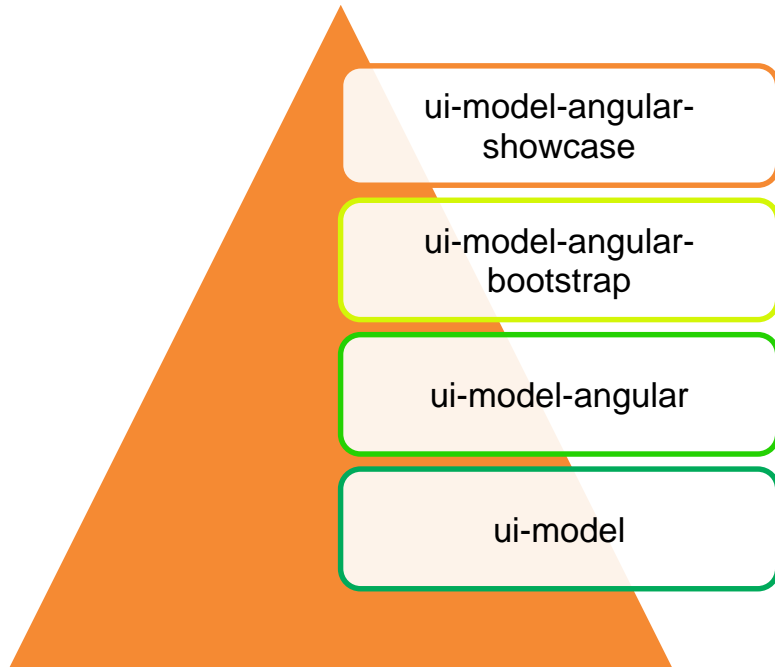
不依赖具体框架

- 只要能在视图和模型之间同步即可
- 不管你用V-DOM实现还是脏检查

可以提供包装层

- 包装层用于项目内复用
- 可以任意定制，出错的可能性很低

ui-model的分层复用模型



代码比例（Calendar为例）

ui-model（独立于框架）

- 205行实现
- 194行测试

Angular

- 37行

React

- 73行

开发团队

技术专家（共享）

工程师

前端JS
框架

CSS

ui-model
复用库

...

全栈开发

思考题

难度1: 滚动数字框 (带动画)



153,9⁴⁴

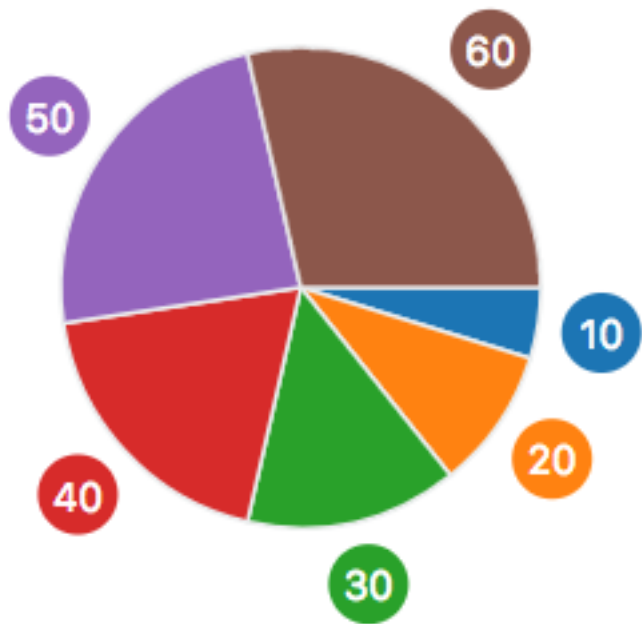
实现数字视口

- 20 ts + 6 html + 4 scss

实现滚动数字框

- 35 ts + 3 html + 12 scss

难度2: 图表



抽象出坐标系 200+40

抽象出图形 220+70

绑定到svg 18

Q & A



ThoughtWorks®