**墙裂推荐大家阅读自定义组件的官方文档**

<https://developers.weixin.qq.com/miniprogram/dev/framework/custom-component/>

首先说一下组件的重要性

比如在我们的微信小程序里，我们可能随处可见这种样式



如果纯写wxml的话，要写4处复杂的代码，分别加上不同的标题，副标题，图片

但是他们的组成元素是一样的，都是由 “标题”，“副标题”，“时间”，“图片”构成的，

那么我们可不可以写一个抽象的组件来表示这个东西呢，**当然能，而且非常必要~！**

我之前不就写了这么两个东西放在components目录下嘛，当然项目伊始，科技树可能没有那么完善。

就在2019年4月23日，**激动人心的事来了，八哥引入了we-ui组件库，这意味着我们能使用别人家写的样式进行页面布局了**，咱们的自行车变成了摩托

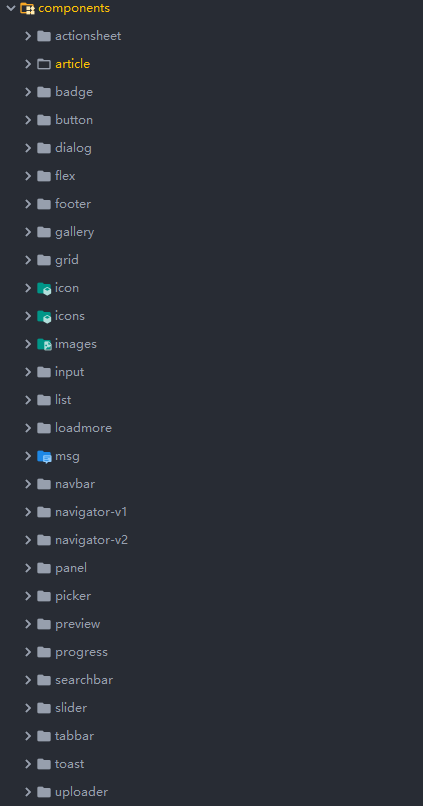
然鹅比较让人桑心的是咱们的摩托是坏的，需要咱们修一哈

于是也许大家需要先完成修车这一步（忧桑）

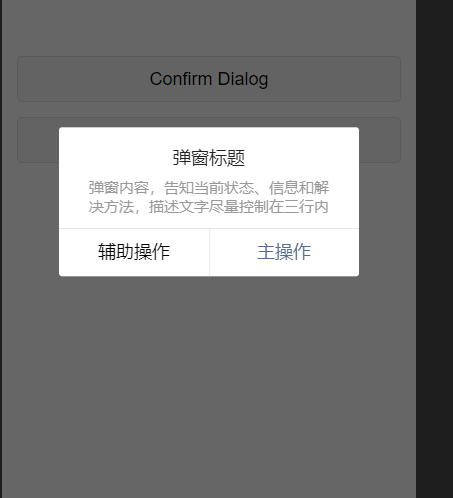
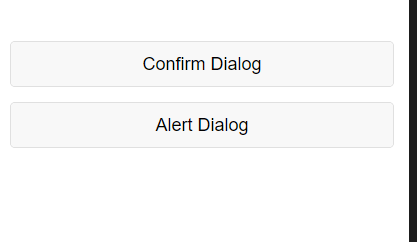
不过这也是个好事，大家在维护的过程中能学到很多东西，之后的页面开发工作可能遇到的阻碍也会越小。

**那么问题来了，这个车咱们该咋修呢**

首先大家更新了代码后，client/components 目录下是这么些玩意儿

balabalabala…这些就是我引入的一堆组件

打开某一个文件夹，比如我们进到dialog文件夹，有我们很熟悉的四个东西 .js,.json,.wxml,.wxss。这是一个点击按钮出现提示窗的组件

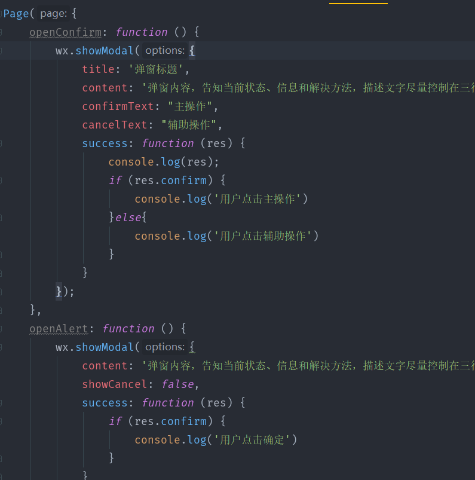
 我们触碰（bindtap）Confirm按钮 ，会出来这么一个弹窗，**这是官方给我们的一个小DEMO，我们现在把它抽象成一个组件**

我们的第一步工作是给组件抽象一些属性

比如在dialog.wxml中，现在的写法是这样的C:\Users\Richard\AppData\Local\Temp\1556019995(1).png

就是我们上面看到的这个Confirm按钮， class是类名， type是button原生自带的属性（**自定义属性时尽量不要与原生的属性重名**）,bindtap是点击事件，里面是按钮的名字，那么我们先抽象这个按钮的名字。

我们打开dialog.js进行编辑。

看到这么一个东西，we-ui给的是个Page({})的对象

### 我们阅读官方文档后，知道组件应该这么写（给大家摘取了一些文档片段）：创建自定义组件

类似于页面，一个自定义组件由 json wxml wxss js 4个文件组成。要编写一个自定义组件，首先需要在 json 文件中进行自定义组件声明（将 component 字段设为 true 可这一组文件设为自定义组件）：

{

"component": true

}

同时，还要在 wxml 文件中编写组件模板，在 wxss 文件中加入组件样式，它们的写法与页面的写法类似。具体细节和注意事项参见 [组件模板和样式](https://developers.weixin.qq.com/miniprogram/dev/framework/custom-component/wxml-wxss.html) 。

**代码示例：**

<!-- 这是自定义组件的内部WXML结构 -->

<view class="inner">

{{innerText}}

</view>

<slot></slot>

/\* 这里的样式只应用于这个自定义组件 \*/

.inner {

color: red;

}

**注意：在组件wxss中不应使用ID选择器、属性选择器和标签名选择器。**

在自定义组件的 js 文件中，需要使用 [Component](https://developers.weixin.qq.com/miniprogram/dev/reference/api/Component.html) 来注册组件，并提供组件的属性定义、内部数据和自定义方法。

组件的属性值和内部数据将被用于组件 wxml 的渲染，其中，属性值是可由组件外部传入的。更多细节参见 [Component构造器](https://developers.weixin.qq.com/miniprogram/dev/framework/custom-component/component.html) 。

**代码示例：**

Component({

properties: {

// 这里定义了innerText属性，属性值可以在组件使用时指定

innerText: {

type: String,

value: 'default value',

}

},

data: {

// 这里是一些组件内部数据

someData: {}

},

methods: {

// 这里是一个自定义方法

customMethod() {}

}

})

使用自定义组件

使用已注册的自定义组件前，首先要在页面的 json 文件中进行引用声明。此时需要提供每个自定义组件的标签名和对应的自定义组件文件路径：

{

"usingComponents": {

"component-tag-name": "path/to/the/custom/component"

}

}

照葫芦画瓢，我们在.json文件下设置“component”:true

在.js文件下, 把Page 改成 Component, **注意这时写法和Page不同了**

Component({

properties:{

自定义属性名：{

type:属性类型

value:默认值

}

}，

data:{

组件内部变量名：值

},

Methods: {

函数方法名(){

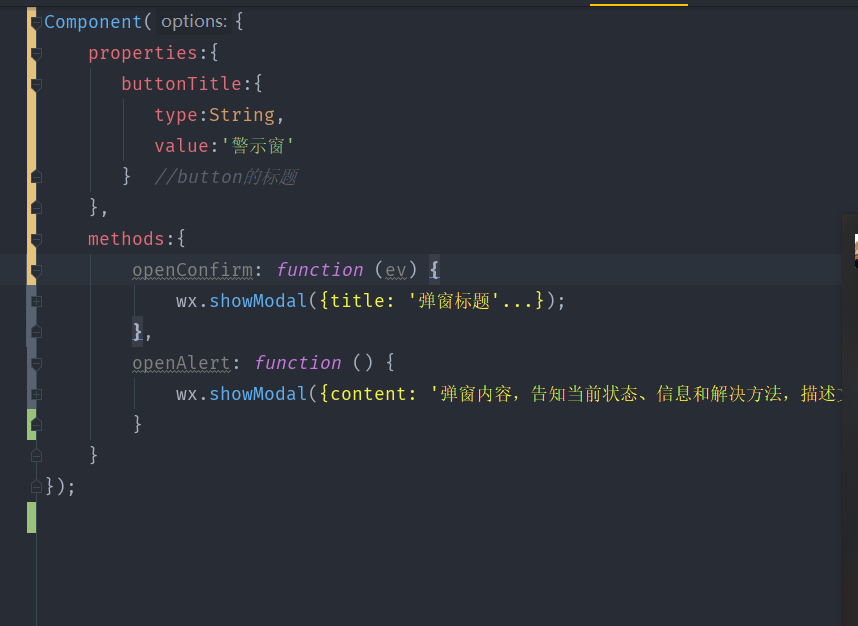
函数逻辑

}

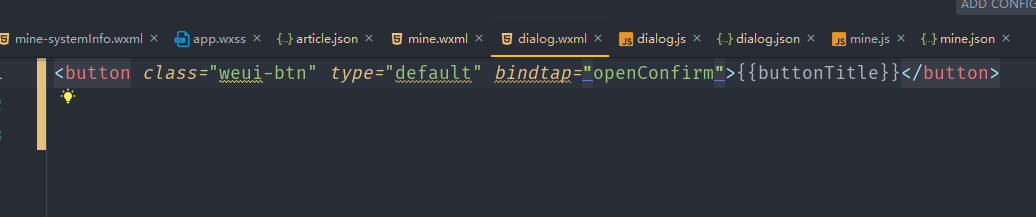
}

})，**依照这个，我们来修改一下，为button添加名字属性**

**代码如下：.js文件**

****

**.wxml文件：**

****

**.json文件上面描述过了**

**这里设置了buttonTitle属性，默认为警示窗，以后我们调用这个组件时，加上buttonTitle即可自行设置按钮标题。**

**接下来我们在页面上调用这个组件**

**随便进入page目录下的一个文件夹**

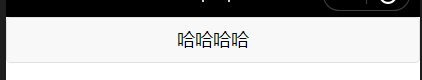
**在.json文件那设置：**

{  
 "usingComponents": {  
 "dialog": "/components/dialog/dialog"  
 }  
}

**“dialog”为自行设定的标签名 ，然后我们就可以在.wxml上用dialog标签来表示这个组件啦，试试吧~~**

<dialog *buttonTitle=*"哈哈哈哈"></dialog>

**页面呈现效果：**

****

**好嘞，咱艰难的跨出了第一步~！**

**接下来我们需要做的工作就是抽象这些组件，可以有一些自己的思考，第一步先将we-ui给的示例代码变成能复用的东西，遇到不懂的可以跟我探讨一哈~我自己也会进行组件的抽象开发，这个工作量还是挺大的，咱先好好分工一下**