

手把手教你实现小程序中的自定义组件



实现小程序自定义组件

手把手教你！实现小程序中的自定义组件

之前做[小程序](#)开发的时候，对于开发来说比较头疼的莫过于自定义组件了，当时官方对这方面的文档也只是寥寥几句，一笔带过而已，所以写起来真的是非常非常痛苦！！

好在微信小程序的库从 1.6.3 开始，官方对于自定义组件这一块有了比较大的变动，首先比较明显的感觉就是文档比以前全多了，有木有！（小程序文档），现在小程序支持简洁的组件化编程，可以将页面内的功能模块抽象成自定义组件，以便在不同的页面中复用，提高自己代码的可读性，降低自己维护代码的成本！

本篇文章就是手把手教你实现小程序中自定义组件，坐稳啦~

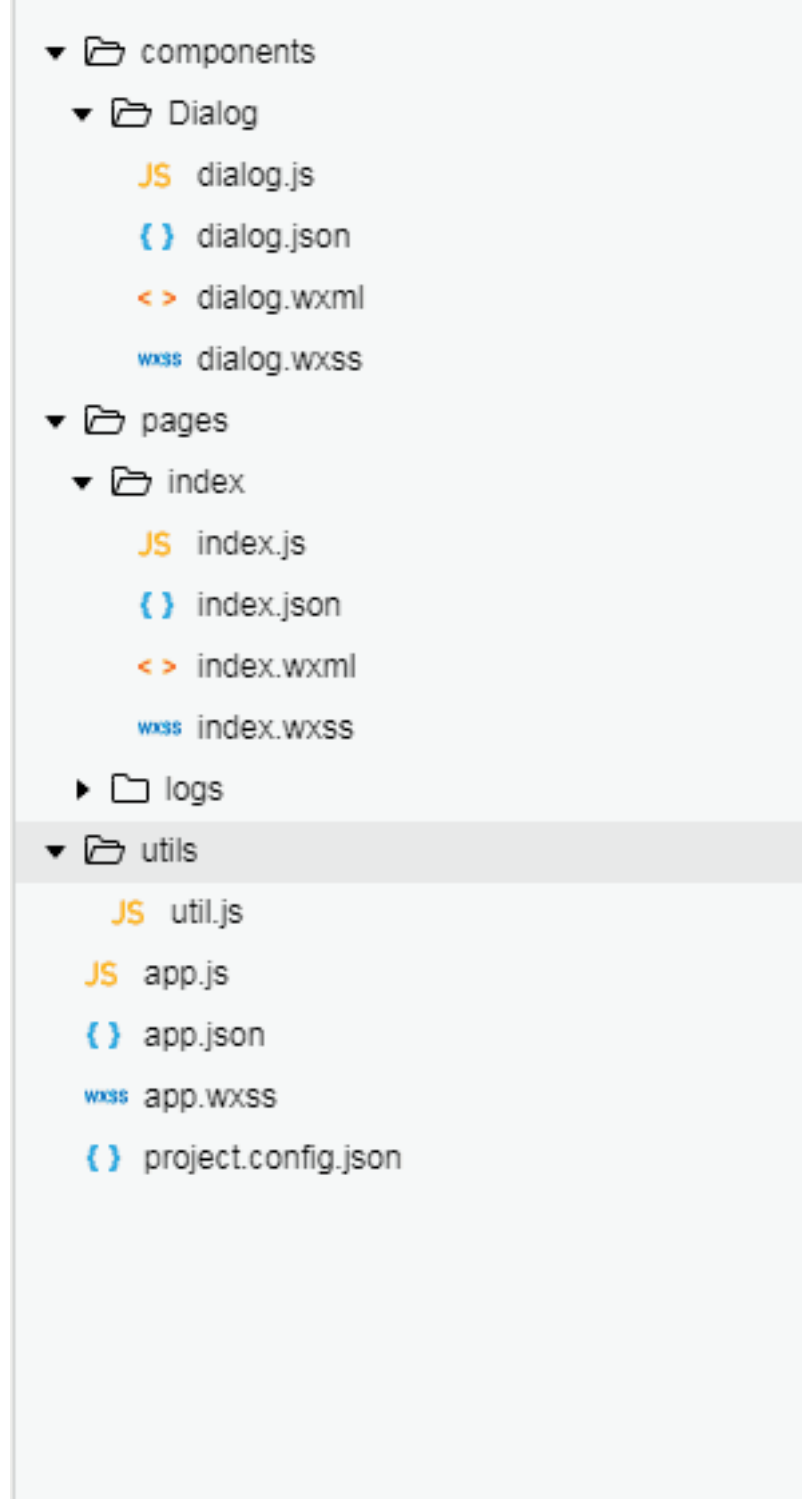
具体实现

要做自定义组件，我们先定一个小目标，比如说我们在小程序中实现一下 WEUI 中的弹窗组件，基本效果图如下。



Step1

我们初始化一个小程序（本示例基础版本库为 1.7），删掉里面的示例代码，并新建一个 **components** 文件夹，用于存放我们以后开发中的所用组件，今天我们的目的是实现一个 弹框 组件，因此，我们在 **components** 组件中新建一个 **Dialog** 文件夹来存放我们的弹窗组件，在 **Dialog** 下右击新建 **Component** 并命名为 **dialog** 后，会生成对应的 **json wxml wxss js** 4 个文件，也就是一个自定义组件的组成部分，此时你的项目结构应该如下图所示：



Step2

组件初始化工作准备完成，接下来就是组件的相关配置，首先我们需要声明自定义组件，也就是将 **dialog.json** 中 **component** 字段设为 **true**：

```
{  
  "component": true,           // 自定义组件声明  
  "usingComponents": {}       // 可选项，用于引用别的组件  
}
```

其次，我们需要在 **dialog.wxml** 文件中编写弹窗组件模版，在 **dialog.wxss** 文件中加入弹窗组件样式，它们的写法与页面的写法类似，我就不赘述，直接贴代码啦~

dialog.wxml 文件如下:

```
<!--components/Dialog/dialog.wxml-->
<view class='wx_dialog_container' hidden="{{!isShow}}">
  <view class='wx-mask'></view>
  <view class='wx-dialog'>
    <view class='wx-dialog-title'>{{ title }}</view>
    <view class='wx-dialog-content'>{{ content }}</view>
    <view class='wx-dialog-footer'>
      <view class='wx-dialog-btn' catchtap='_cancelEvent'>{{ cancelText }}</view>
      <view class='wx-dialog-btn' catchtap='_confirmEvent'>{{ confirmText }}</view>
    </view>
  </view>
</view>
```

dialog.wxss 文件如下:

```
/* components/Dialog/dialog.wxss */
.wx-mask{
  position: fixed;
  z-index: 1000;
  top: 0;
  right: 0;
  left: 0;
  bottom: 0;
  background: rgba(0, 0, 0, 0.3);
}
.wx-dialog{
  position: fixed;
  z-index: 5000;
  width: 80%;
  max-width: 600rpx;
  top: 50%;
  left: 50%;
  -webkit-transform: translate(-50%, -50%);
  transform: translate(-50%, -50%);
  background-color: #FFFFFF;
  text-align: center;
  border-radius: 3px;
  overflow: hidden;
}
.wx-dialog-title{
  font-size: 18px;
  padding: 15px 15px 5px;
}
```

```
.wx-dialog-content{
  padding: 15px 15px 5px;
  min-height: 40px;
  font-size: 16px;
  line-height: 1.3;
  word-wrap: break-word;
  word-break: break-all;
  color: #999999;
}

.wx-dialog-footer{
  display: flex;
  align-items: center;
  position: relative;
  line-height: 45px;
  font-size: 17px;
}

.wx-dialog-footer::before{
  content: '';
  position: absolute;
  left: 0;
  top: 0;
  right: 0;
  height: 1px;
  border-top: 1px solid #D5D5D6;
  color: #D5D5D6;
  -webkit-transform-origin: 0 0;
  transform-origin: 0 0;
  -webkit-transform: scaleY(0.5);
  transform: scaleY(0.5);
}

.wx-dialog-btn{
  display: block;
  -webkit-flex: 1;
  flex: 1;
  -webkit-tap-highlight-color: rgba(0, 0, 0, 0);
  position: relative;
}

.wx-dialog-footer .wx-dialog-btn:nth-of-type(1){
  color: #353535;
}

.wx-dialog-footer .wx-dialog-btn:nth-of-type(2){
  color: #3CC51F;
}

.wx-dialog-footer .wx-dialog-btn:nth-of-type(2):after{
  content: " ";
  position: absolute;
  left: 0;
  top: 0;
```

```
width: 1px;
bottom: 0;
border-left: 1px solid #D5D5D6;
color: #D5D5D6;
-webkit-transform-origin: 0 0;
transform-origin: 0 0;
-webkit-transform: scaleX(0.5);
transform: scaleX(0.5);
}
```

step3

组件的结构和样式都有了，还缺少什么呢，没错，还缺 **js**，眼睛比较犀利的同学，可能已经发现了我们在 **dialog.wxml** 文件中的会有一些比如 **{{ isShow }}**、**{{ title }}** 这样的模版变量，还定义了 **_cancelEvent** 和 **_confirmEvent** 两个方法，其具体实现就是在 **dialog.js** 中。

dialog.js 是自定义组件的构造器，是使用小程序中 **Component** 构造器生成的，调用 **Component** 构造器时可以用来指定自定义组件的属性、数据、方法等，具体的细节可以参考一下官方的文档

下面我通过代码注释解释一下构造器中的一些属性的使用：

```
// components/Dialog/dialog.js
Component({
  options: {
    multipleSlots: true // 在组件定义时的选项中启用多slot支持
  },
  /**
   * 组件的属性列表
   * 用于组件自定义设置
   */
  properties: {
    // 弹窗标题
    title: { // 属性名
      type: String, // 类型（必填），目前接受的类型包括：String, Number, Boo
      value: '标题' // 属性初始值（可选），如果未指定则会根据类型选择一个
    },
    // 弹窗内容
    content: {
      type: String ,
```

```

        value : '弹窗内容'
    },
    // 弹窗取消按钮文字
    cancelText :{
        type : String ,
        value : '取消'
    },
    // 弹窗确认按钮文字
    confirmText :{
        type : String ,
        value : '确定'
    }
},

/**
 * 私有数据,组件的初始数据
 * 可用于模版渲染
 */
data: {
    // 弹窗显示控制
    isShow:false
},

/**
 * 组件的方法列表
 * 更新属性和数据的方法与更新页面数据的方法类似
 */
methods: {
    /*
     * 公有方法
     */

    //隐藏弹框
    hideDialog(){
        this.setData({
            isShow: !this.data.isShow
        })
    },
    //展示弹框
    showDialog(){
        this.setData({
            isShow: !this.data.isShow
        })
    },
    /*
     * 内部私有方法建议以下划线开头
     * triggerEvent 用于触发事件
     */

```

```
    _cancelEvent(){
      //触发取消回调
      this.triggerEvent("cancelEvent")
    },
    _confirmEvent(){
      //触发成功回调
      this.triggerEvent("confirmEvent");
    }
  }
})
```

step4

截至目前为止，你应该完成了一个自定义弹窗组件的大部分，可是你保存后并没有发现任何变化，因为我们还需要在 **index.wxml** 文件中引入它！

首先需要在 **index.json** 中引入组件：

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

然后我们在 **index.wxml** 中引入它，并增加我们自定义的一些值，如下

```
<!--index.wxml-->
<view class="container">
  <dialog id='dialog'
    title='我是标题'
    content='恭喜你，学会了小程序组件'
    cancelText='知道了'
    confirm='谢谢你'
    bind:cancelEvent="_cancelEvent"
    bind:confirmEvent="_confirmEvent">
  </dialog>

  <button type="primary" bindtap="showDialog"> ClickMe! </button>
</view>
```


嗯哪，还差最后一步， `index.js` 配置，没错，这个也很简单，我就复制粘贴了

```
//index.js
//获取应用实例
const app = getApp()

Page({

  /**
   * 生命周期函数--监听页面初次渲染完成
   */
  onReady: function () {
    //获得dialog组件
    this.dialog = this.selectComponent("#dialog");
  },


  showDialog(){
    this.dialog.showDialog();
  },

  //取消事件
  _cancelEvent(){
    console.log('你点击了取消');
    this.dialog.hideDialog();
  },
  //确认事件
  _confirmEvent(){
    console.log('你点击了确定');
    this.dialog.hideDialog();
  }
})
```

到此！大功告成！

step5

让我们测试一下试试看：

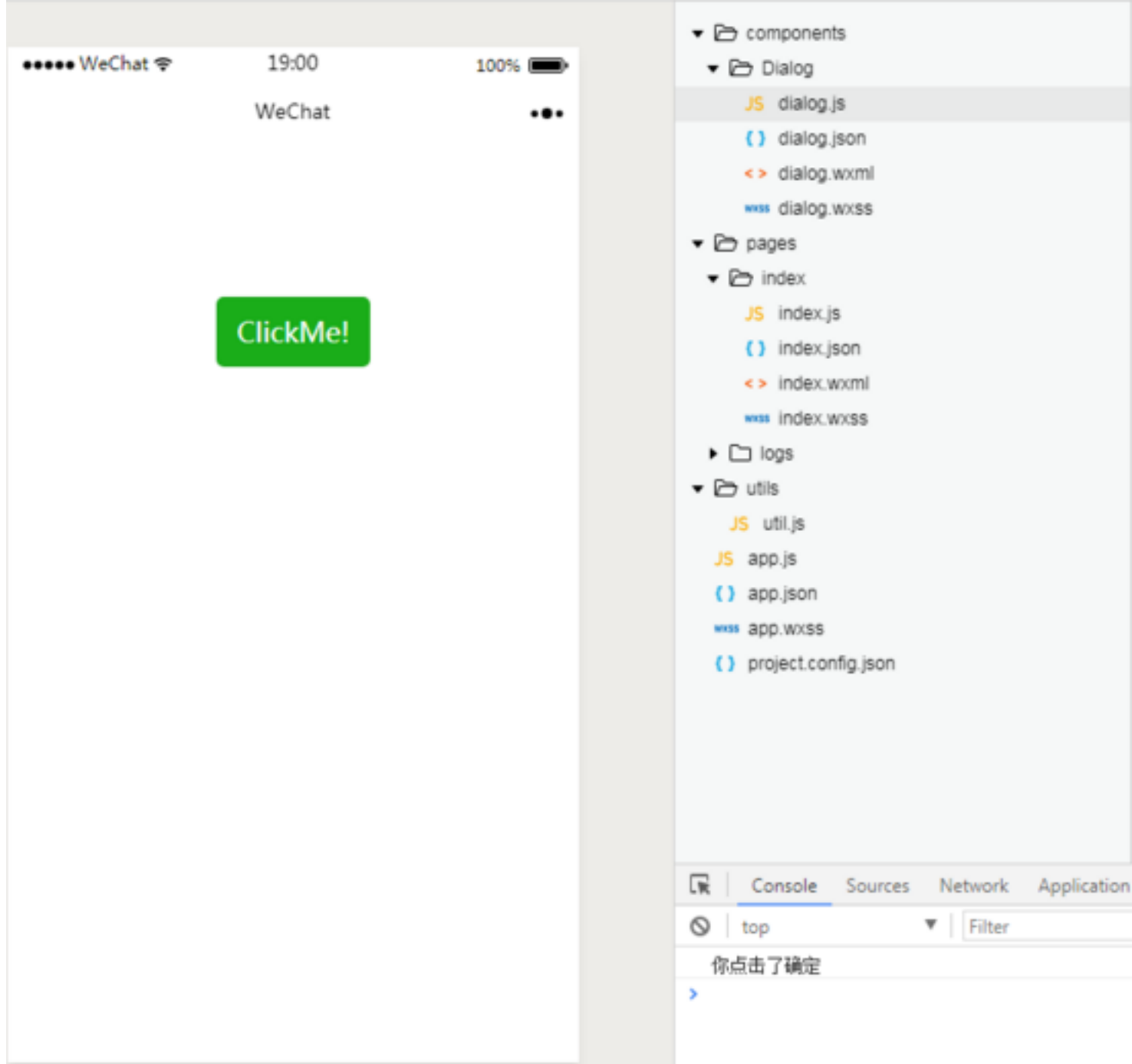


ClickMe!

点击按钮之后呢，会出现如下效果：



点击取消或者确定按钮的话，我们在事件中设置了弹窗会关闭，并会打印出相应的信息，具体点击完应该怎么做，就看你们自己发挥了，我只能帮你到这里了~



总结

现在，你已经基本掌握了小程序中的自定义组件开发技巧，怎么样，是不是很棒，应该给自己点个赞，打个call。总体来说，小程序推出自定义组件后，感觉方便了很多，还没有 get 的小伙伴们，赶紧学习学习，以后多用组件化开发，就不会那么难受了，加油哦~

具体代码，我已经托管到 [github](#) 上了，欢迎 **issue** ~

本文作者：Chris

原文地址：[手把手教你实现微信小程序中的自定义组件-教程-小程序社区-微信小程序-微信小程序开发社区-小程序开发论坛-微信小程序联盟](#)