# 微信小程序\_02

# 今日目标6

* 能够给view组件绑定tap事件
* 能够给事件传递参数 data-
* 能会使用scroll-view和view容器组件
* 能够使用按钮进行分享功能实现
* 能够使用wxs模块来过滤输出显示
* 自定义一个组件

# 一、事件

## 1.1、事件绑定

[**https://developers.weixin.qq.com/miniprogram/dev/framework/view/wxml/event.html**](https://developers.weixin.qq.com/miniprogram/dev/framework/view/wxml/event.html)

小程序事件绑定是在组件中以属性的形式来完成的，例【bind/catch事件名="方法名"】。

注：小程序中绑定方法名不能用括号，直接写方法名即可。

* 事件名bind【冒泡】或catch【非冒泡】开头，然后跟上事件的类型，如bindtap、catchtap。

tap 手指触摸后马上离开 类似于 html中的click事件

* 自基础库版本 1.5.0 起，在非原生组件中，bind和catch后可以紧跟一个冒号，其含义不变，如bind:tap、catch:touchstart。

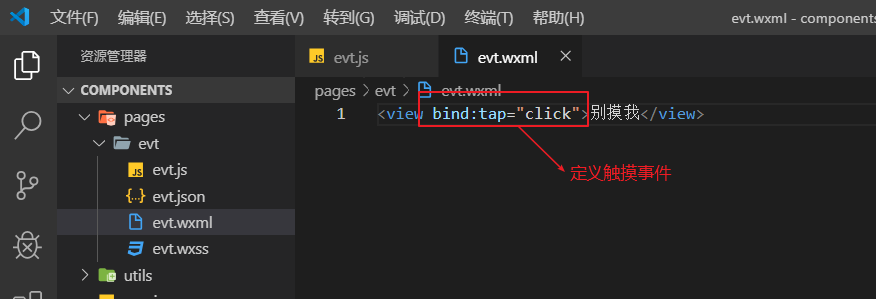
原生组件【camera、canvas、input、live-player、live-pusher、map、textarea、

video】

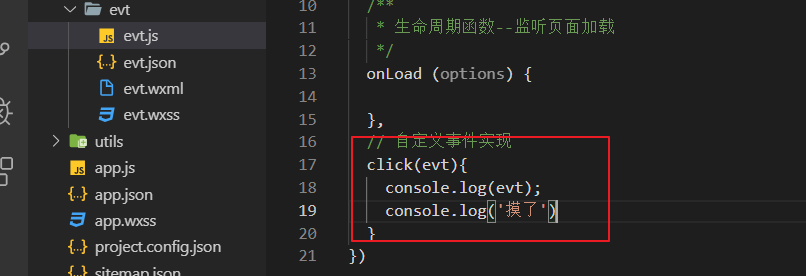
* value 是一个字符串，需要在对应的页面逻辑文件JS文件中定义对应的实现方法。

在wxml中定义一个事件

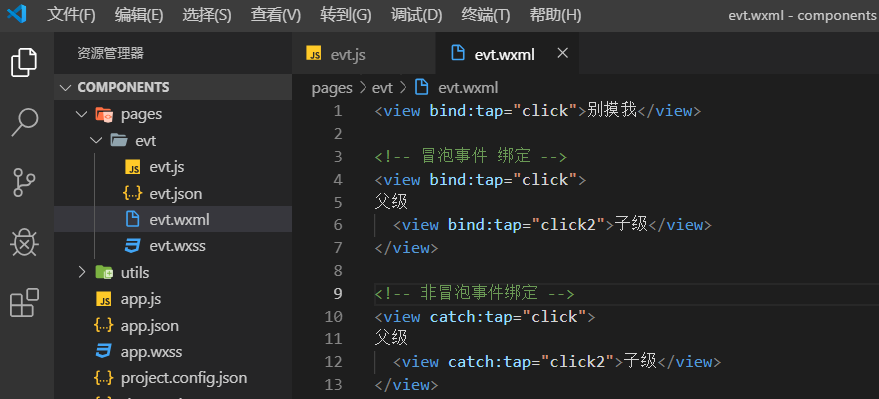
key=value bind:tap=”click” 注意，事件名不能用括号



js中实现对应的事件方法



事件绑定



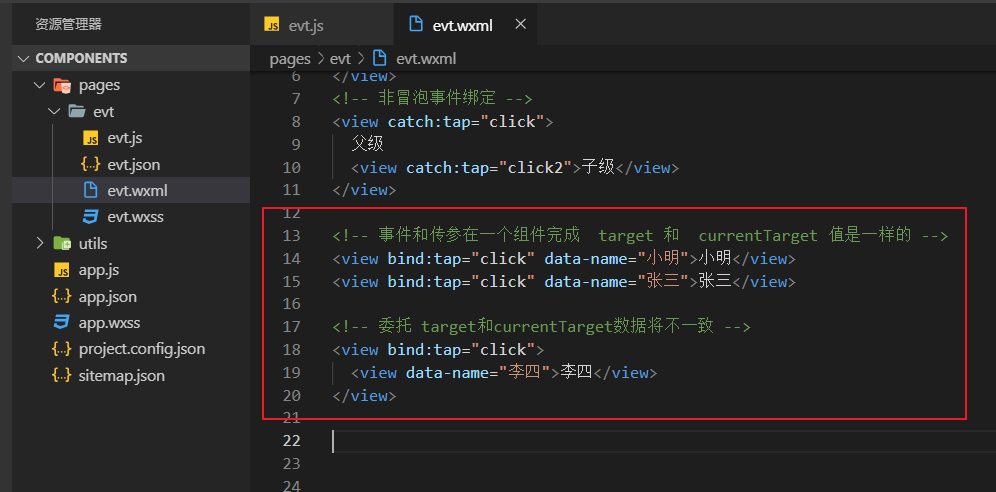
## 1.2、事件对象

如无特殊说明，当组件触发事件时，逻辑层绑定该事件的处理函数会收到一个事件对象。

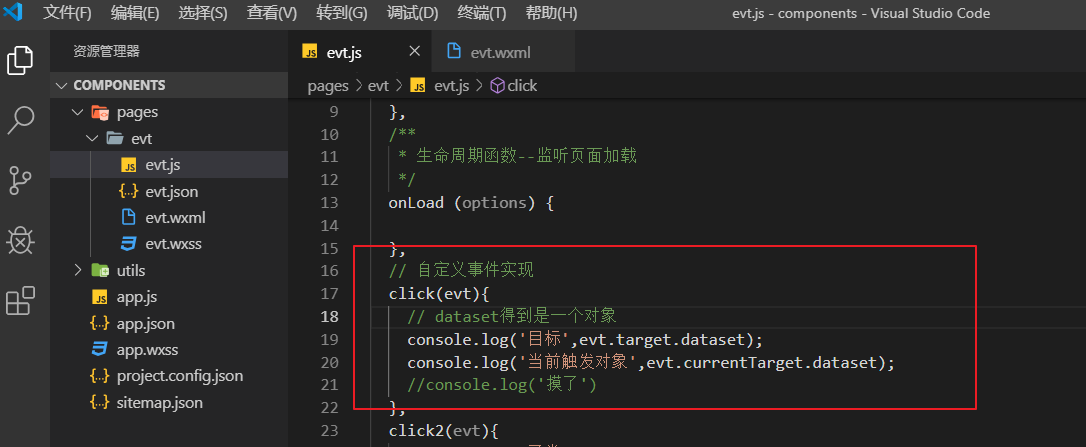
BaseEvent 基础事件对象属性列表：

|  |  |  |
| --- | --- | --- |
| 属性 | 类型 | 说明 |
| type | String | 事件类型 |
| timeStamp | Integer | 事件生成时的时间戳 |
| target | Object | 触发事件的组件的一些属性值集合 |
| currentTarget | Object | 当前组件的一些属性值集合(对象) 在组件中 **data-** 开头定义的属性 |

使用data-开头的的自定义属性来进行参数的传递



在js中通过 event对象中的 target或currentTarget中的dataset来获取



# 二、常用小程序组件

## 2.1、概念及语法

框架为开发者提供了一系列基础组件，开发者可以通过组合这些基础组件进行快速开发。

什么是组件：

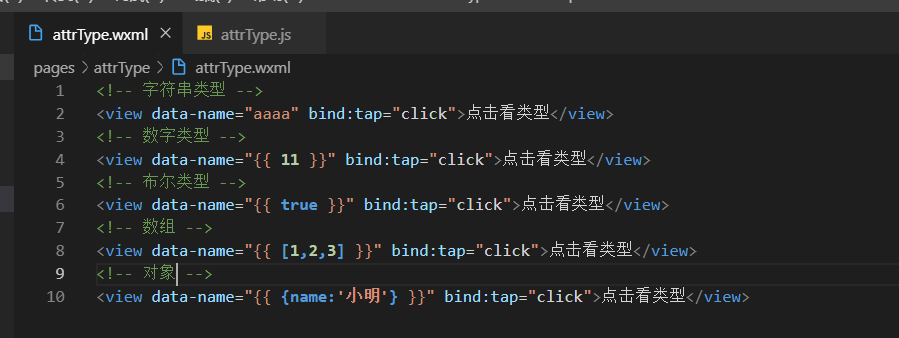
* 组件是视图层的基本组成单元。
* 一个组件通常包括开始标签和结束标签，属性用来修饰这个组件，内容在两个标签之内。

注意：所有组件与**属性**都是**小写**，连接字符用**中横线**【-】连接

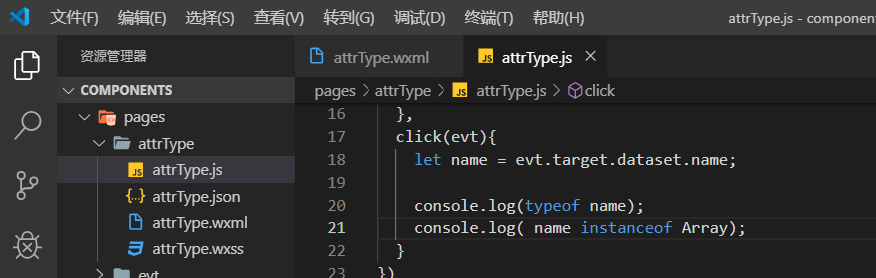
### 2.1.1、属性值类型

|  |  |  |
| --- | --- | --- |
| 类型 | 描述 | 注解 |
| Boolean | 布尔值 | 组件写上该属性，不管该属性等于什么，其值都为true，只有组件上没有写该属性时，属性值才为false。如果属性值为变量，变量的值会被转换为Boolean类型{{}} |
| Number | 数字 | 1, 2.5{{}} |
| String | 字符串 | "string" |
| Array | 数组 | [ 1, "string" ]{{}} |
| Object | 对象 | { key: value }{{}} |
| EventHandler | 事件处理函数名 | "handlerName" 是 Page中定义的事件处理函数名 |

wxml中属性类型传递



js类型判断



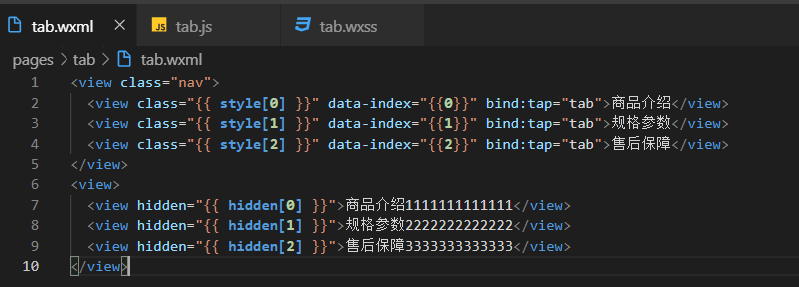
### 2.1.2、共同属性类型

所有组件都有的属性：

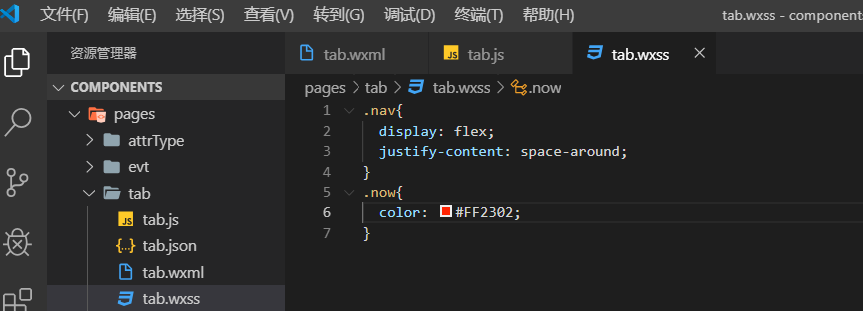
|  |  |  |  |
| --- | --- | --- | --- |
| 属性名 | 类型 | 描述 | 注解 |
| id | String | 组件的唯一标示 | 保持整个页面唯一 |
| class | String | 组件的样式类 | 在对应的 WXSS 中定义的样式类 |
| style | String | 组件的内联样式 | 可以动态设置的内联样式 |
| hidden | Boolean [false] | 组件是否显示 | 所有组件默认显示 |
| data-\* | Any | 自定义属性 | 组件上触发的事件时，会发送给事件处理函数 |
| bind\* / catch\* | EventHandler | 组件的事件 | 详见事件 |

tab切换

在wxml中定义对应的数据和事件

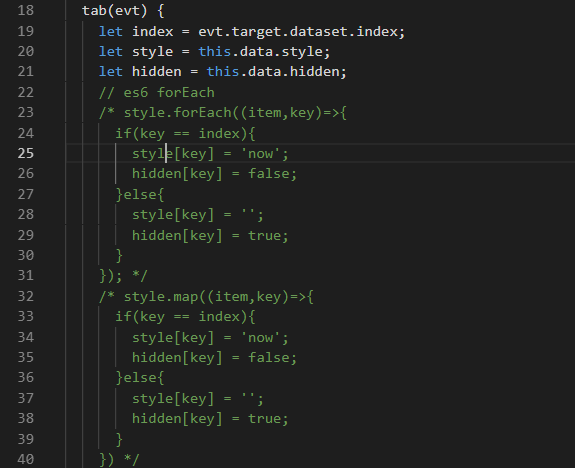


wxss样式



js文件







效果

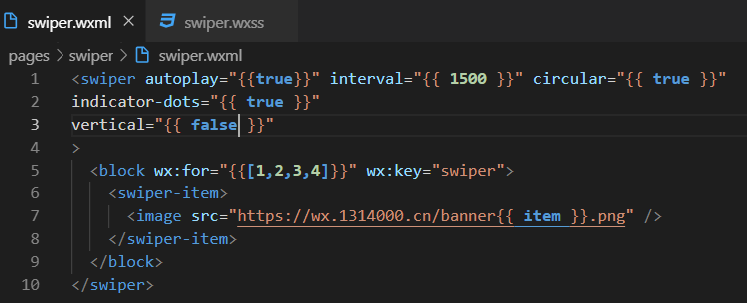


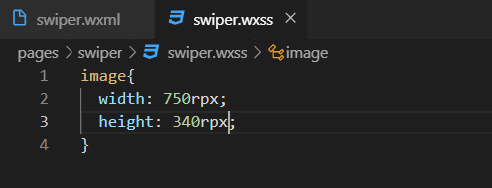
## 2.2、视图容器组件

### 2.2.1、swiper

滑块视图容器。

<https://developers.weixin.qq.com/miniprogram/dev/component/swiper.html>

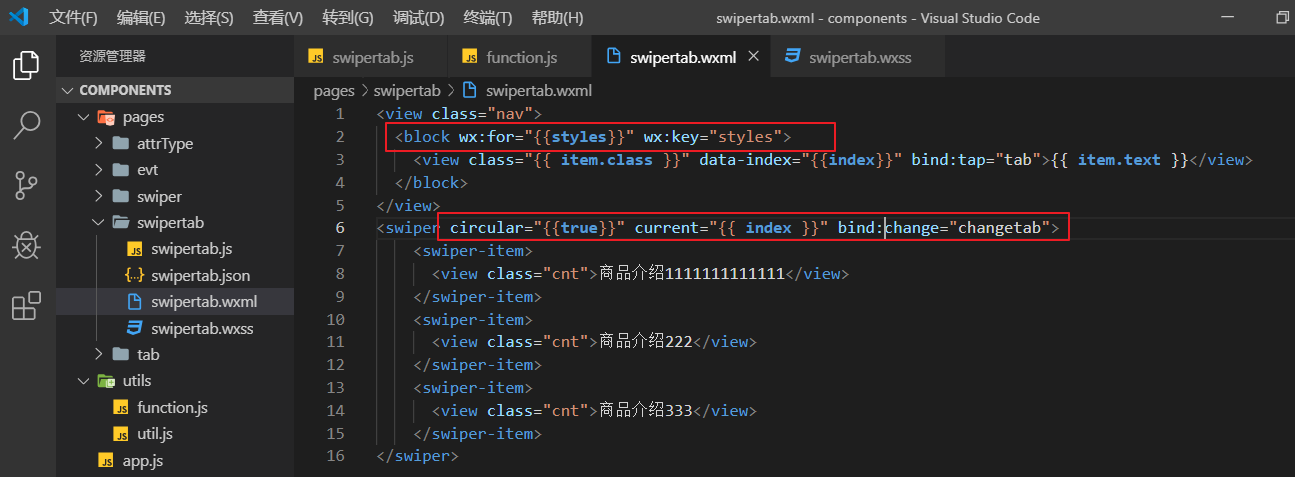




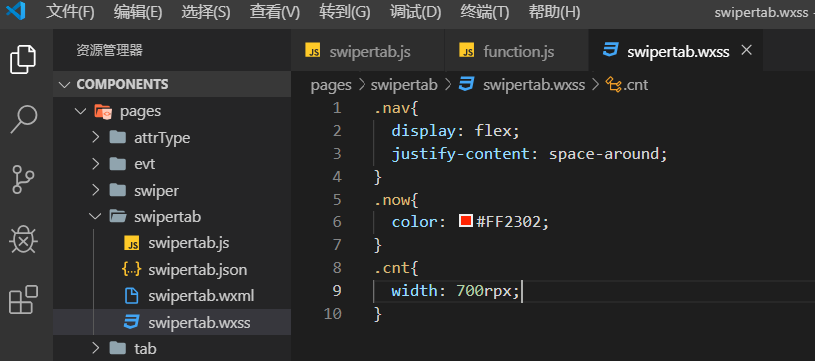


tab案例

wxml循环数据和swiper



wxss定义样式



js中实现swiper的tab切换

使用了小程序提供的模块化功能

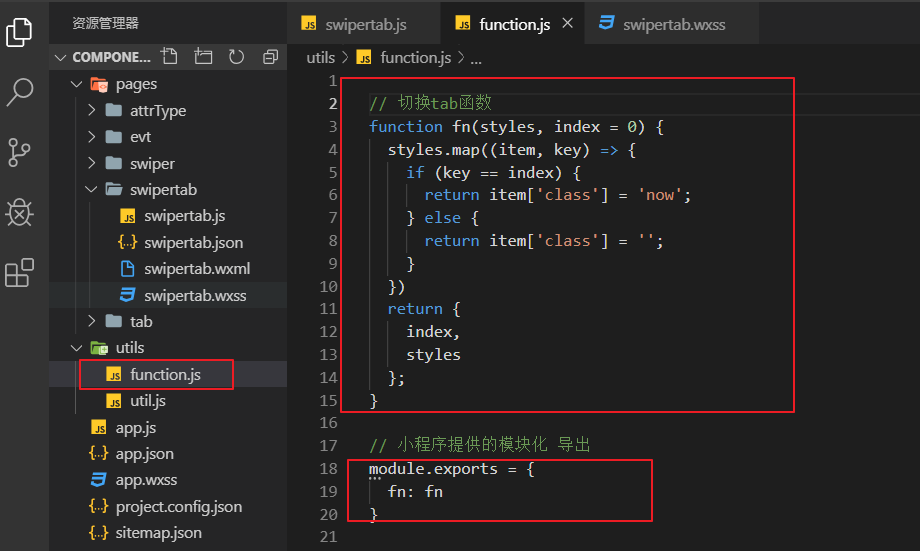
导出

module.exports = {对象}

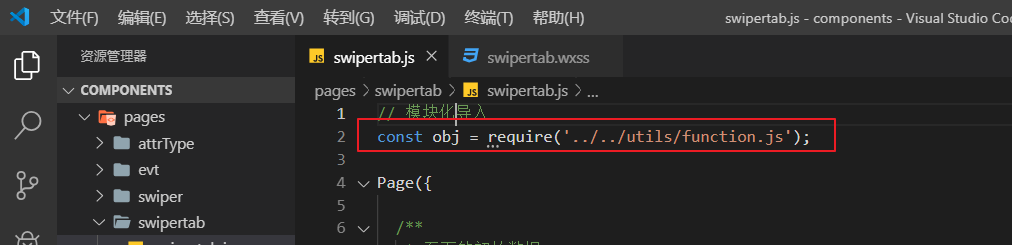
导入

var/let/const 变量 = require(相对地址)

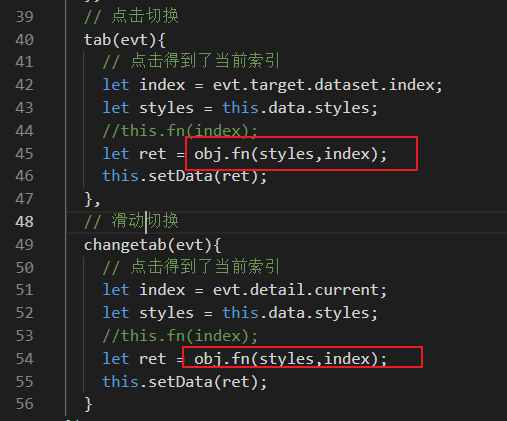
定义公共函数



在js逻辑层中导入引用







### 2.2.2、scroll-view

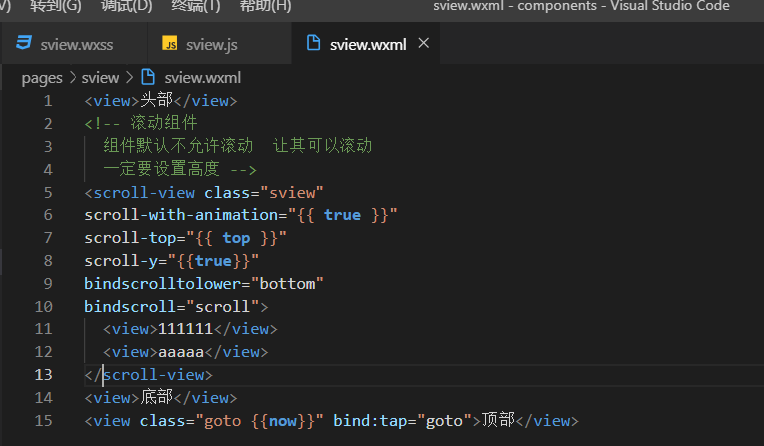
可滚动视图区域。使用竖向滚动时，需要给<scroll-view>一个固定高度，通过 WXSS 设置 height，同时还需要在组件属性中定义允许纵向滚动 scroll-y=”{{true}}”。

注意点

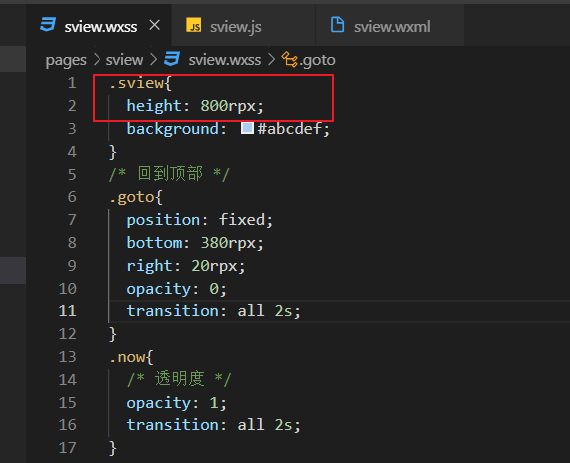
tip: 请勿在 scroll-view 中使用 textarea、map、canvas、video 组件

一定要设置高度，事件触发，必须是大于了你设置的高度才能被触发到事件

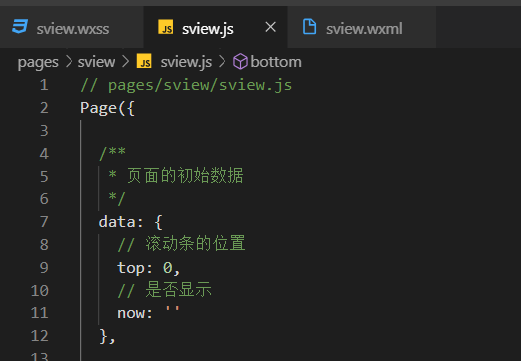
wxml定义滚动组件



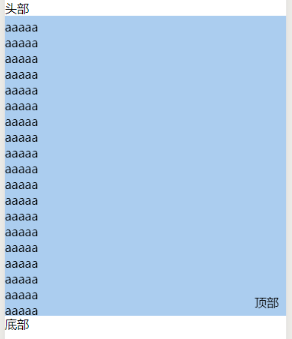
wxss中设置高度



js中实现相关的事件和回到顶部

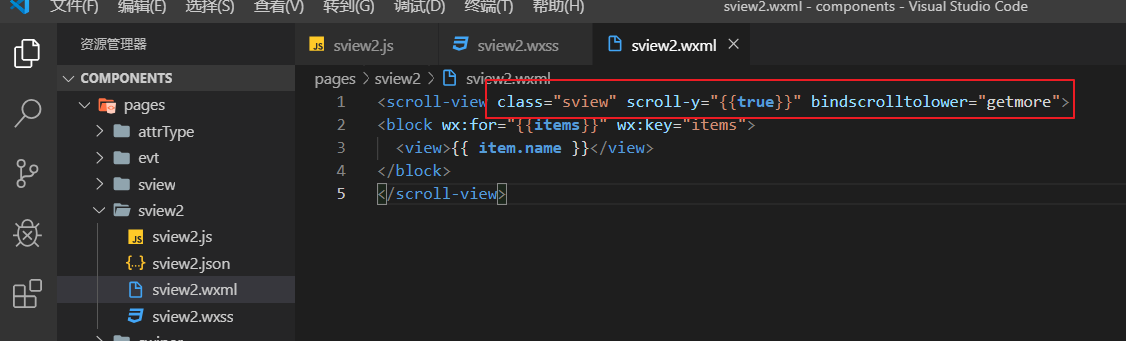




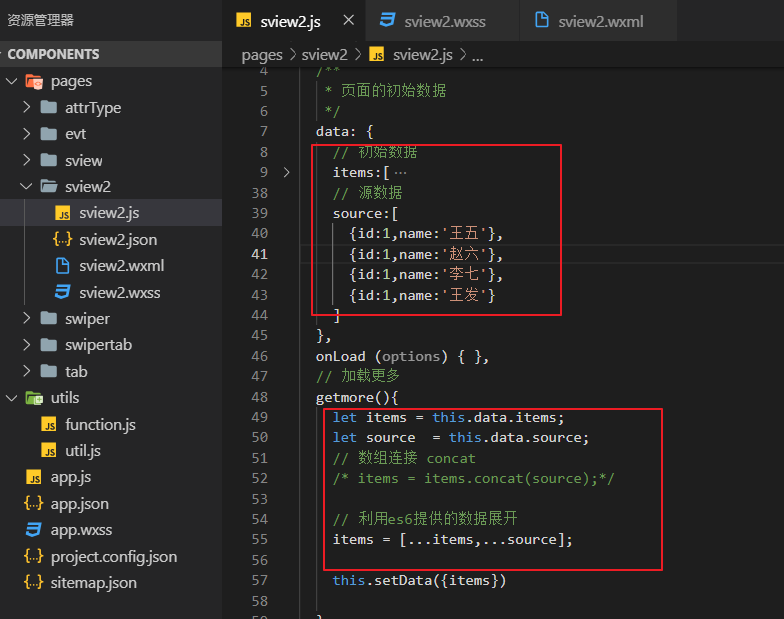


* 加载更多

wxml中循环出数据



js中实现加载更多

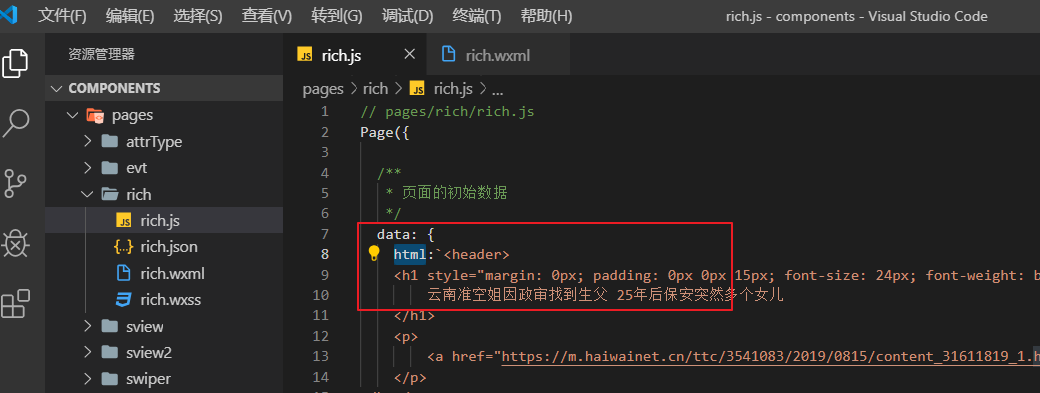


## 2.3、基础内容组件

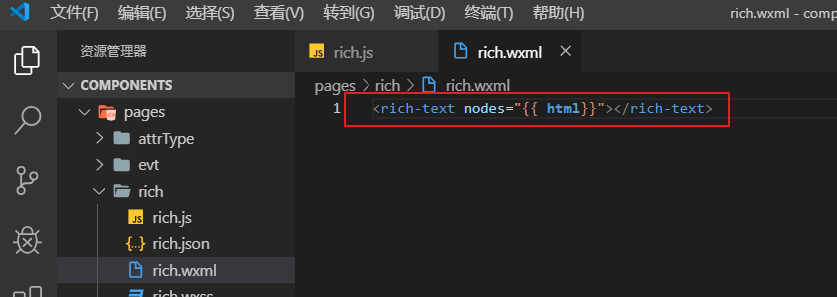
### 2.3.1、rich-text

富文本显示组件

js中定义一个html文本



使用组件显示富文本



但是发现，图片有裁切情况，所以一般项目中不用

#### 2.3.1.1、wxparse 第3方组件

显示富文本

<https://github.com/icindy/wxParse>

下载对应的源码

git clone https://github.com/icindy/wxParse.git

引入必要文件

// 导入到js逻辑文件中

var WxParse = require('../../wxParse/wxParse.js');

// 导入必须的样式文件

@import "/wxParse/wxParse.wxss";

数据绑定

var articleHtml = '<div>我是HTML代码</div>';

/\*\*

\* WxParse.wxParse(bindName , type, data, target,imagePadding)

\* 1.bindName绑定的数据名(必填) 在wxml中调用的名称

\* 2.type可以为html或者md(必填)

\* 3.data为传入的具体数据(必填)

\* 4.target为Page对象,一般为this(必填)

\* 5.imagePadding为当图片自适应是左右的单一padding(默认为0,可选)

\*/

WxParse.wxParse('article', 'html', articleHtml, this, 5);

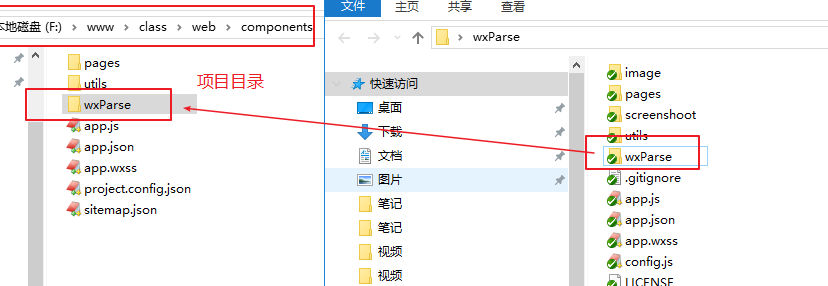
引入模板

<import src="/wxParse/wxParse.wxml"/>

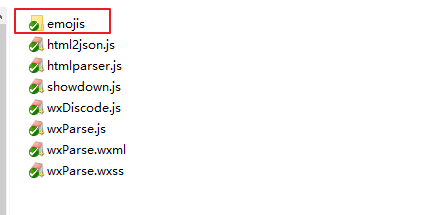
//这里data中article为bindName

<template is="wxParse" data="{{wxParseData:article.nodes}}"/>

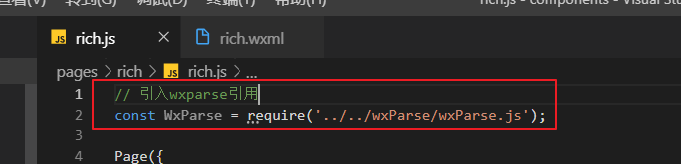
复制相关源码到项目目录中



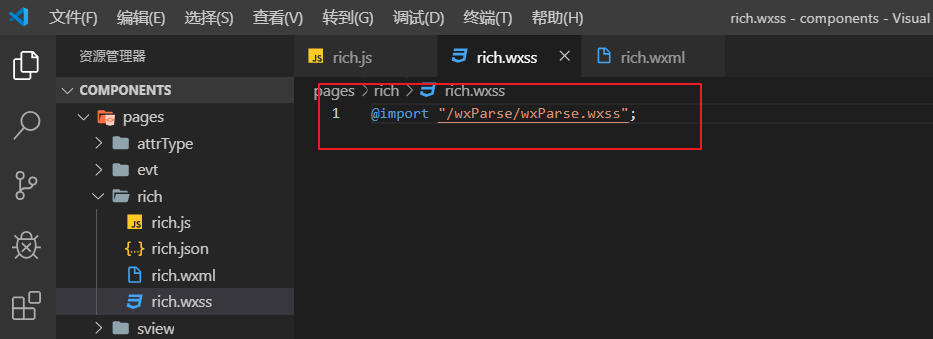
删除无用的emojis



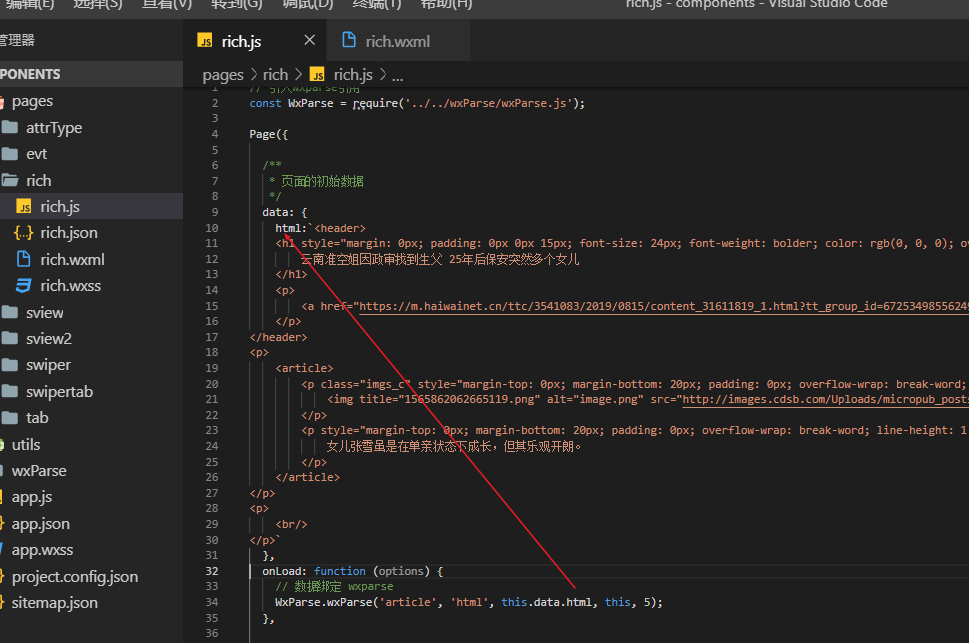
js引用



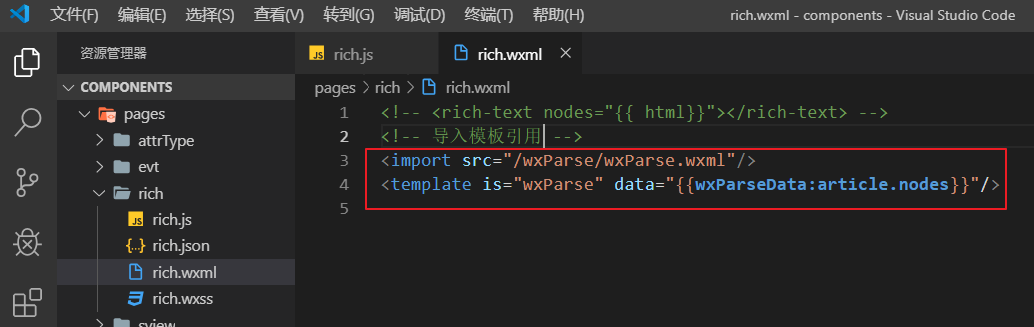
导入样式



js中实现数组绑定



模板引用





优化

注释掉wxparse插件中的console.log输出 html2json.js 和 wxParse.js文件

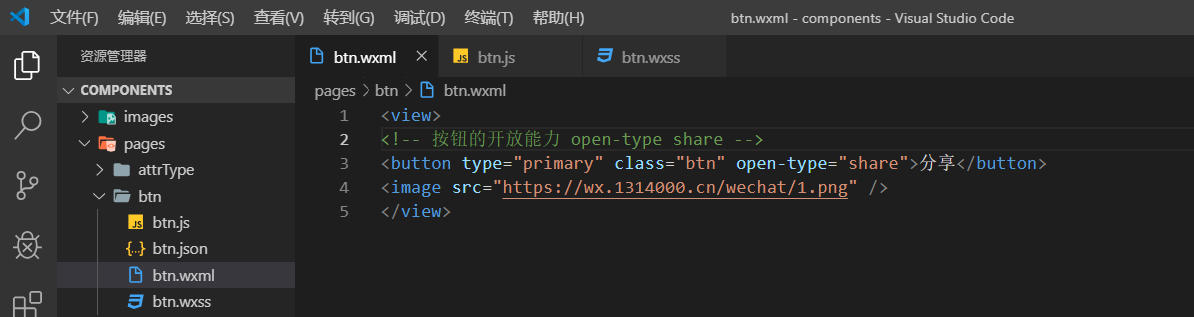
## 2.4、表单组件

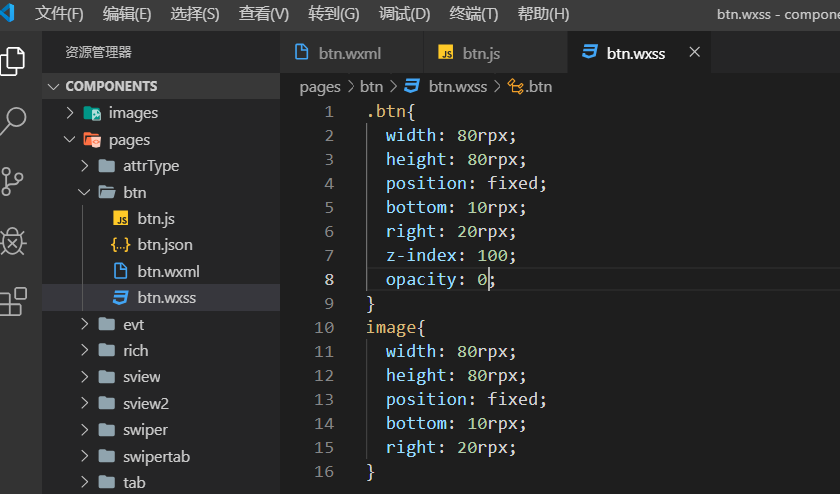
### 2.4.1、按钮 button

<https://developers.weixin.qq.com/miniprogram/dev/component/button.html>

自定义分享功能的实现，开发能力 open-type=”share”

wxml中实现自定义分享能力





js指定自定义分享



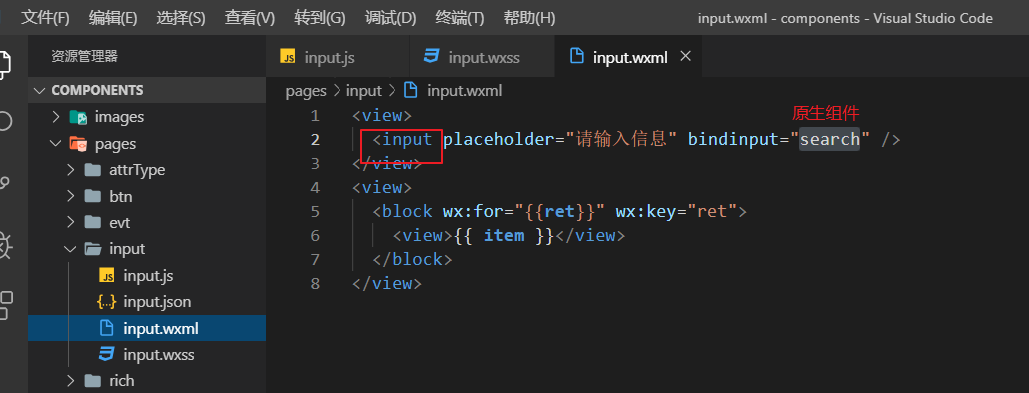
 

### 2.4.2、input

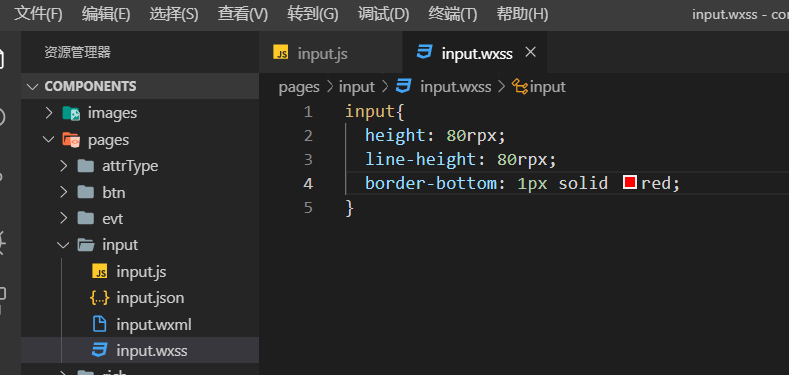
<https://developers.weixin.qq.com/miniprogram/dev/component/input.html>

联想搜索功能的实现

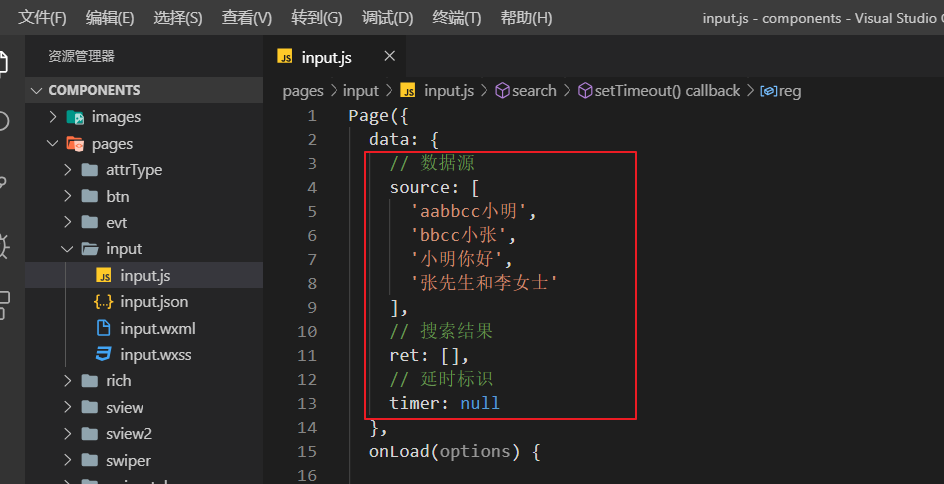
wxml

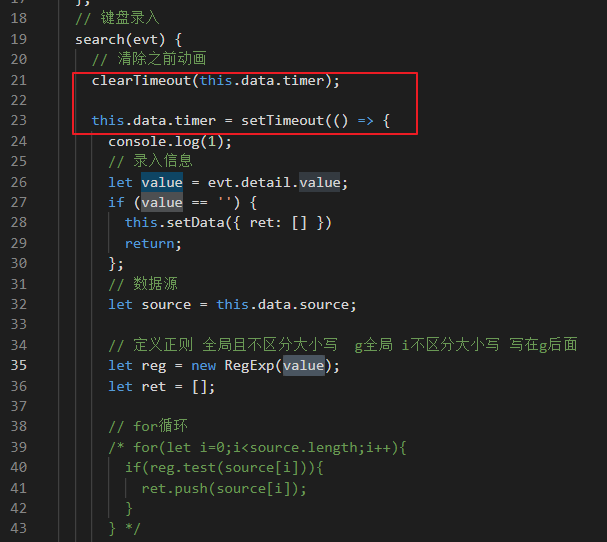


wxss



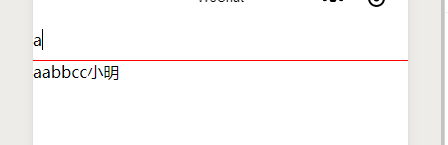
js







效果

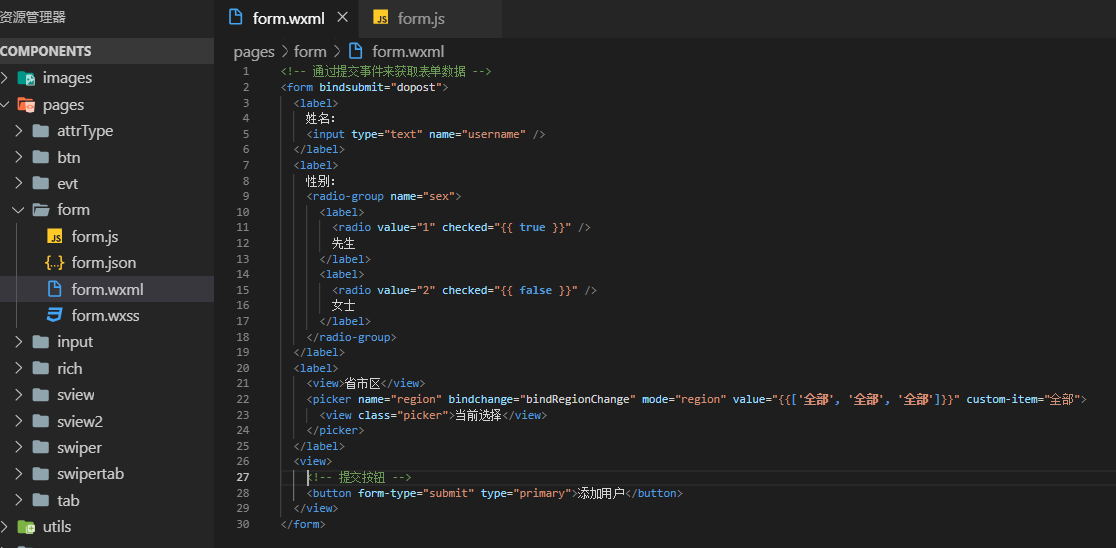


### 2.4.3、form使用

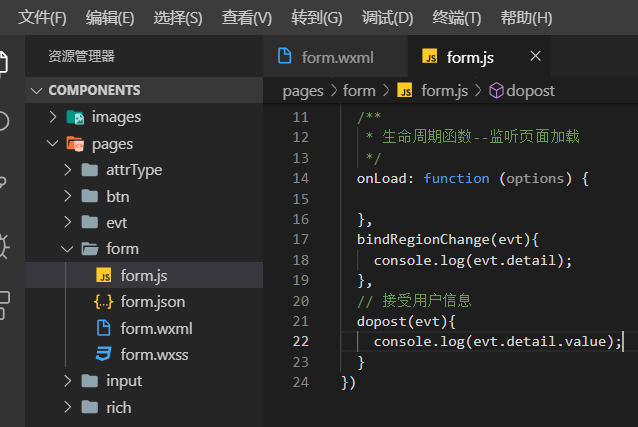
<https://developers.weixin.qq.com/miniprogram/dev/component/form.html>

表单组件，当点击 <form> 表单中 form-type 为 submit 的 <button> 组件时，会将表单组件中的 value 值进行提交，通过onsubmit事件来接受提交过来的信息值，注意的是需要在表单组件中加上 name属性。

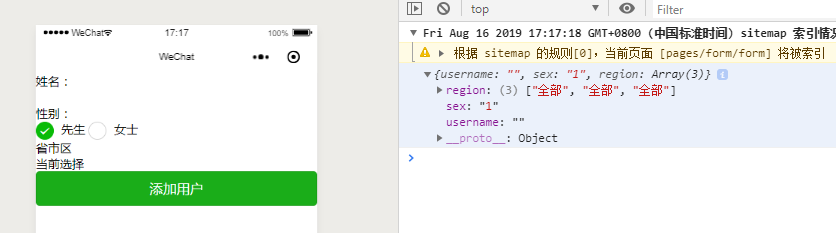
wxml添加用户的表单



js获取数据



效果

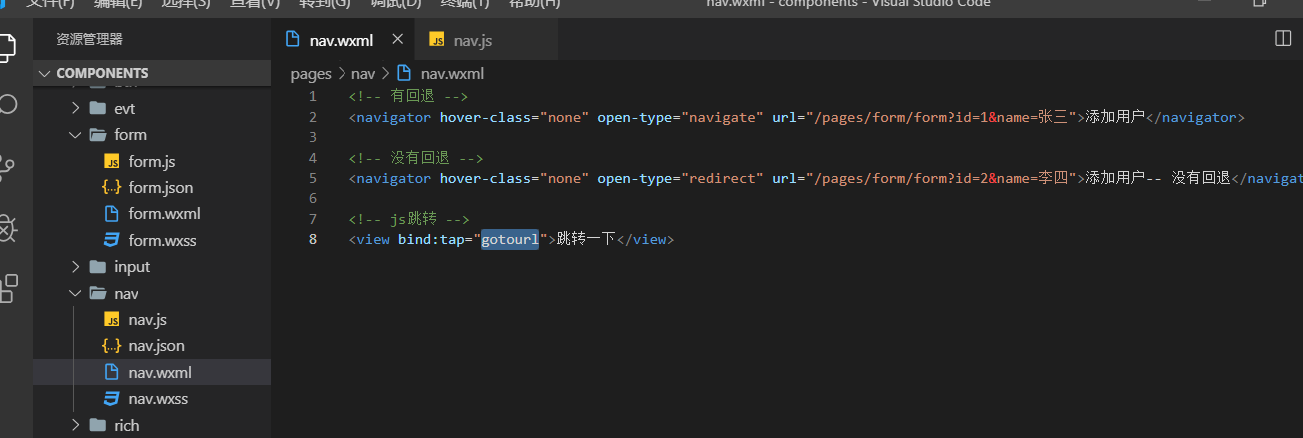


## 2.5、导航

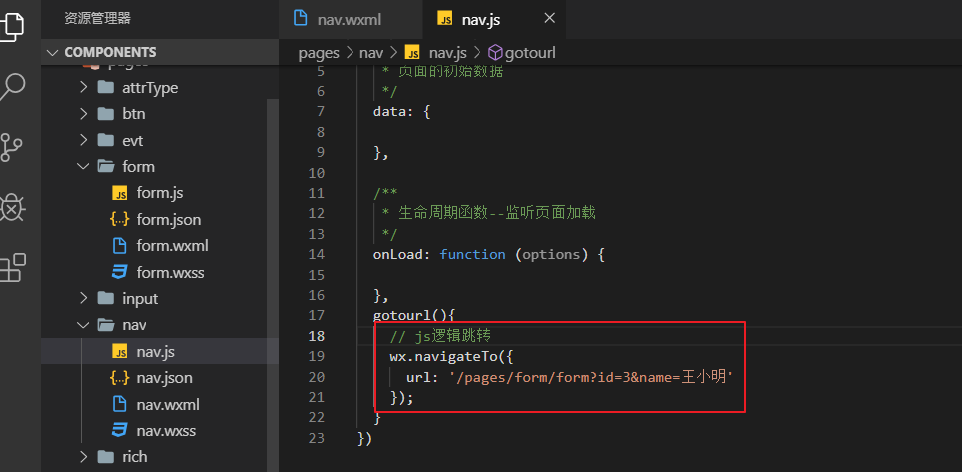
navigator 页面链接。 就是和html中的a标签一样的功能



wxml组件实现跳转



js实现跳转



# 三、自定义组件

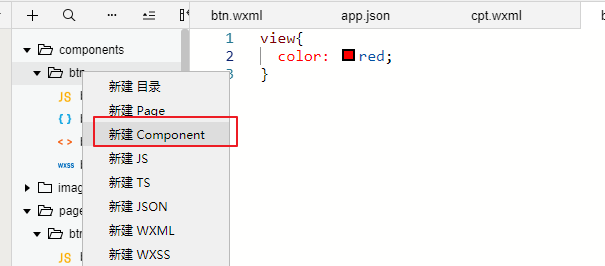
<https://developers.weixin.qq.com/miniprogram/dev/framework/custom-component/>

开发者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重复使用；也可以将复杂的页面拆分成多个低耦合的模块，有助于代码维护。自定义组件在使用时与基础组件非常相似。

说白了就是自定义一些标签，然后这些标签能帮助我们实现一些功能且可以重复的使用。

## 3.1、创建自定义组件

打开官方开发工具，自定义目录后右击菜单，选择如下





## 3.2、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": {

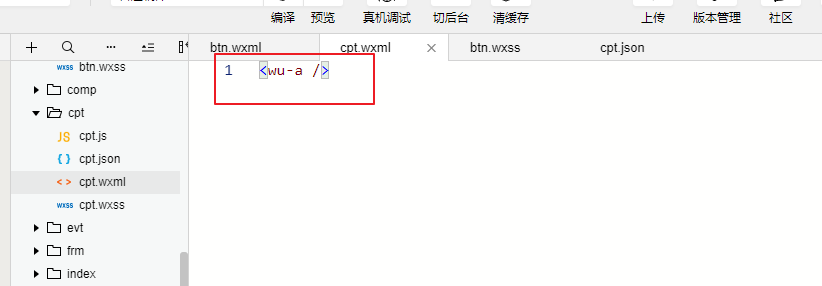
// 在父模板中调用的标签名 组件的路径

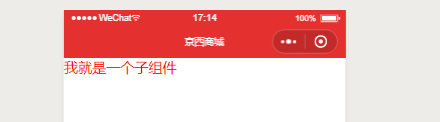
"component-tag-name": "path/to/the/custom/component"

}

}







# 四、WXS 模块

<https://developers.weixin.qq.com/miniprogram/dev/framework/view/wxs/01wxs-module.html>

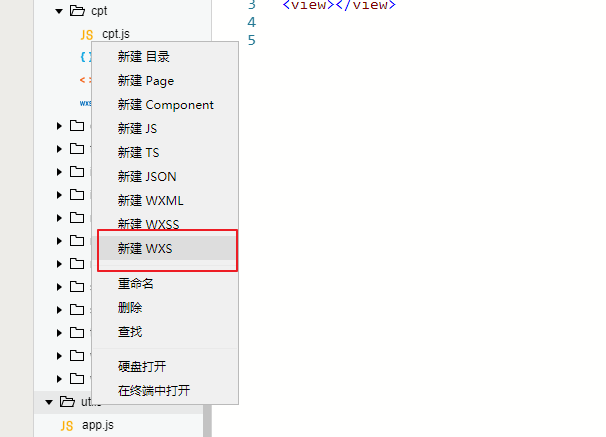
## 4.1、概述

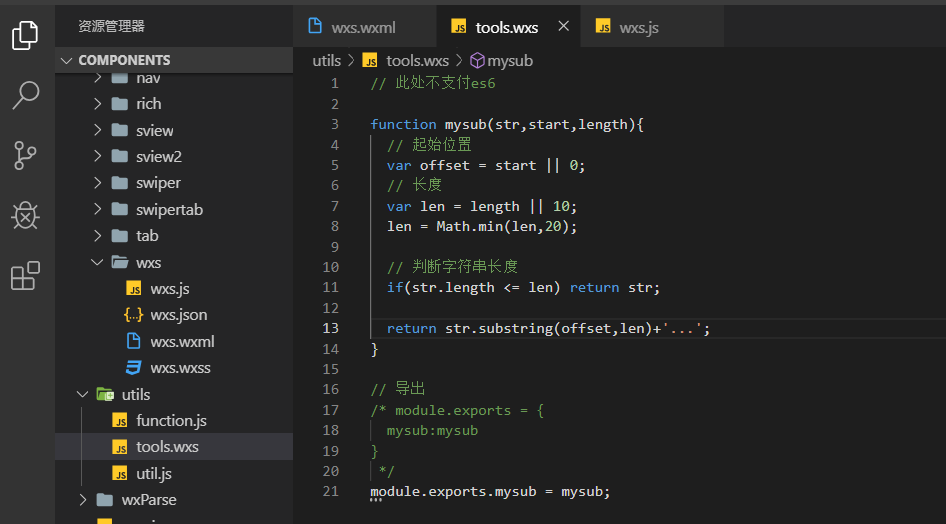
WXS（WeiXin Script）是小程序的一套脚本语言，结合 WXML，可以构建出页面的结构。

WXS中定义的函数可以在wxml文件中使用，可以用它来当过滤器使用。

WXS以.wxs扩展名结尾，文件中代码写法和JS类似，不支持ES6写法

## 4.2、创建wxs模块





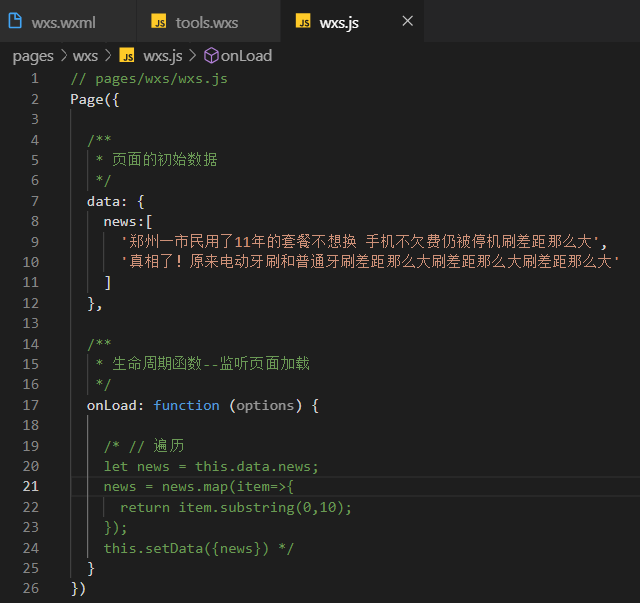
## 4.3、调用

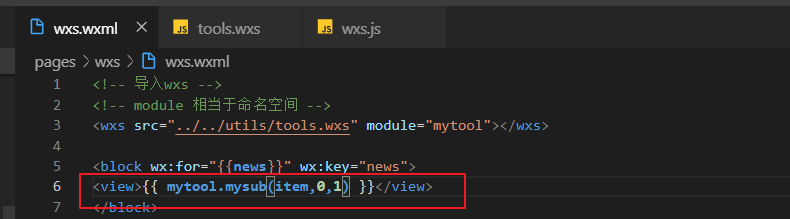
在wxml文件去引用

引用的时候一定要用相对应地址

每个 wxs 模块均有一个内置的 module 对象。 命名空间

<wxs src="../../tools.wxs" module="tools" />







注：wxs和js不相通