

[Component构造器 | 文档 \(qq.com\)](#)

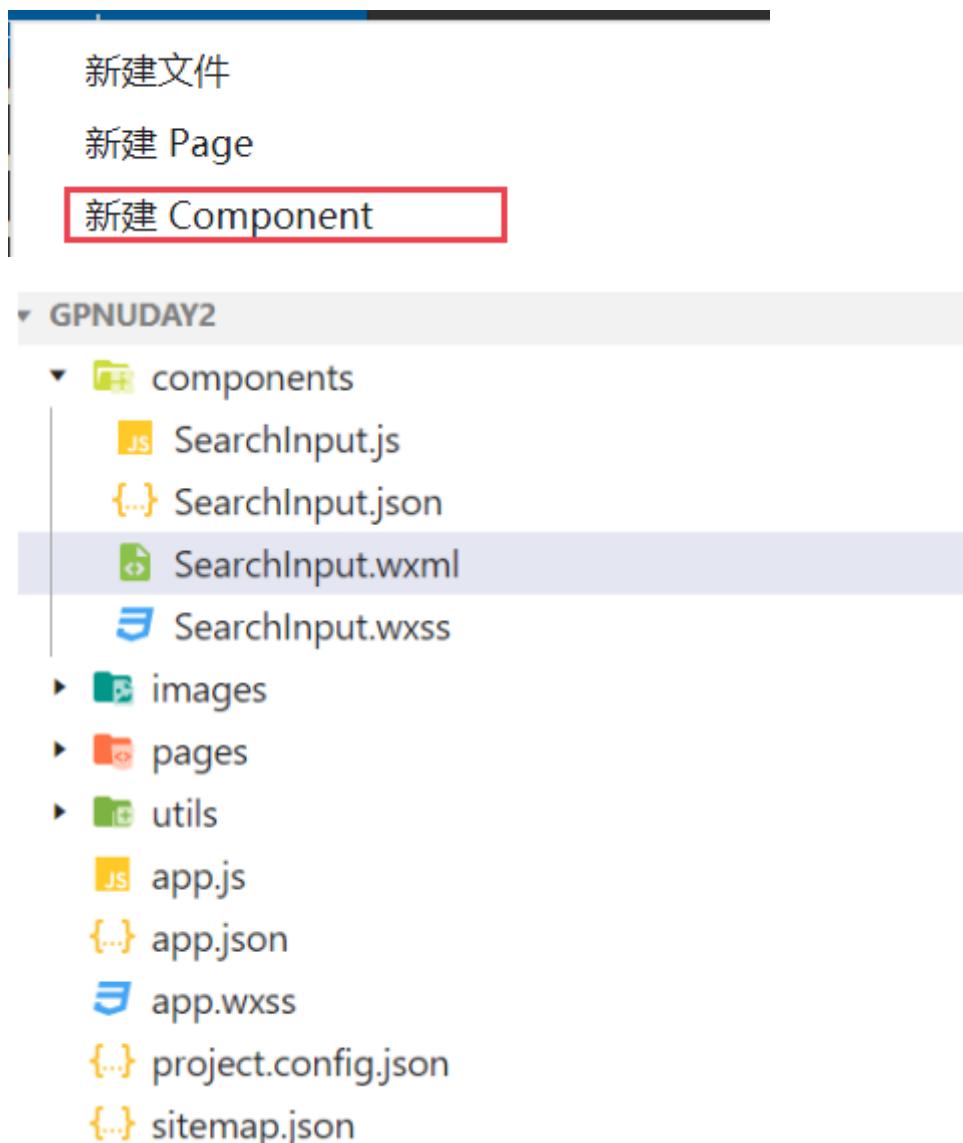
[组件生命周期 | 微信开放文档 \(qq.com\)](#)

[组件模板和样式 | 微信开放文档 \(qq.com\)](#)

[Component 构造器 | 微信开放文档 \(qq.com\)](#)

1.新建自定义组件文件

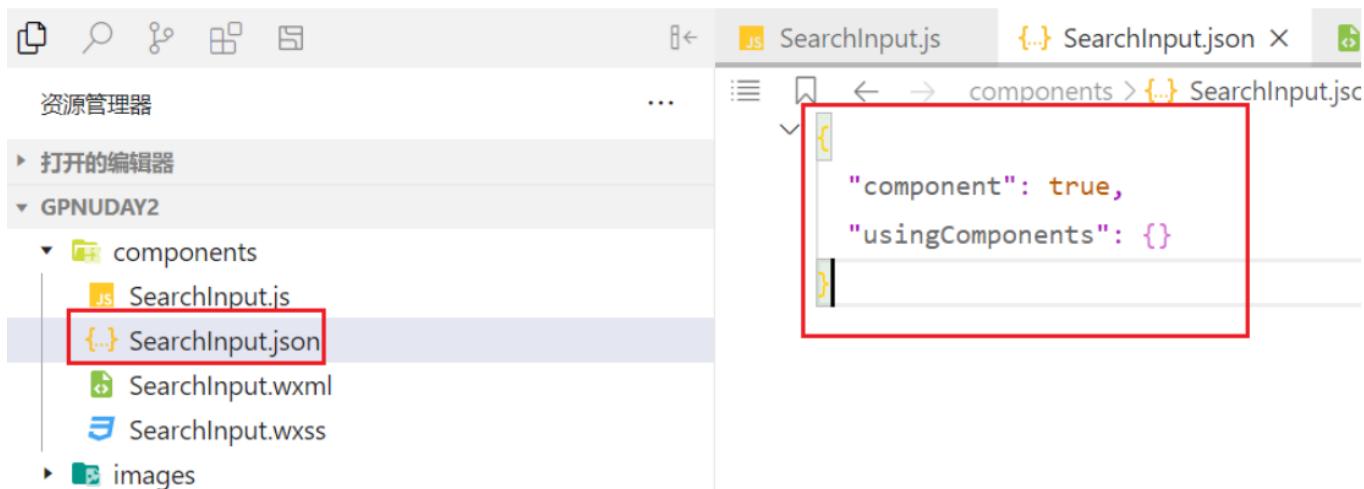
在根目录新建 components 目录，然后右键新建自定义组件目录 SearchInput



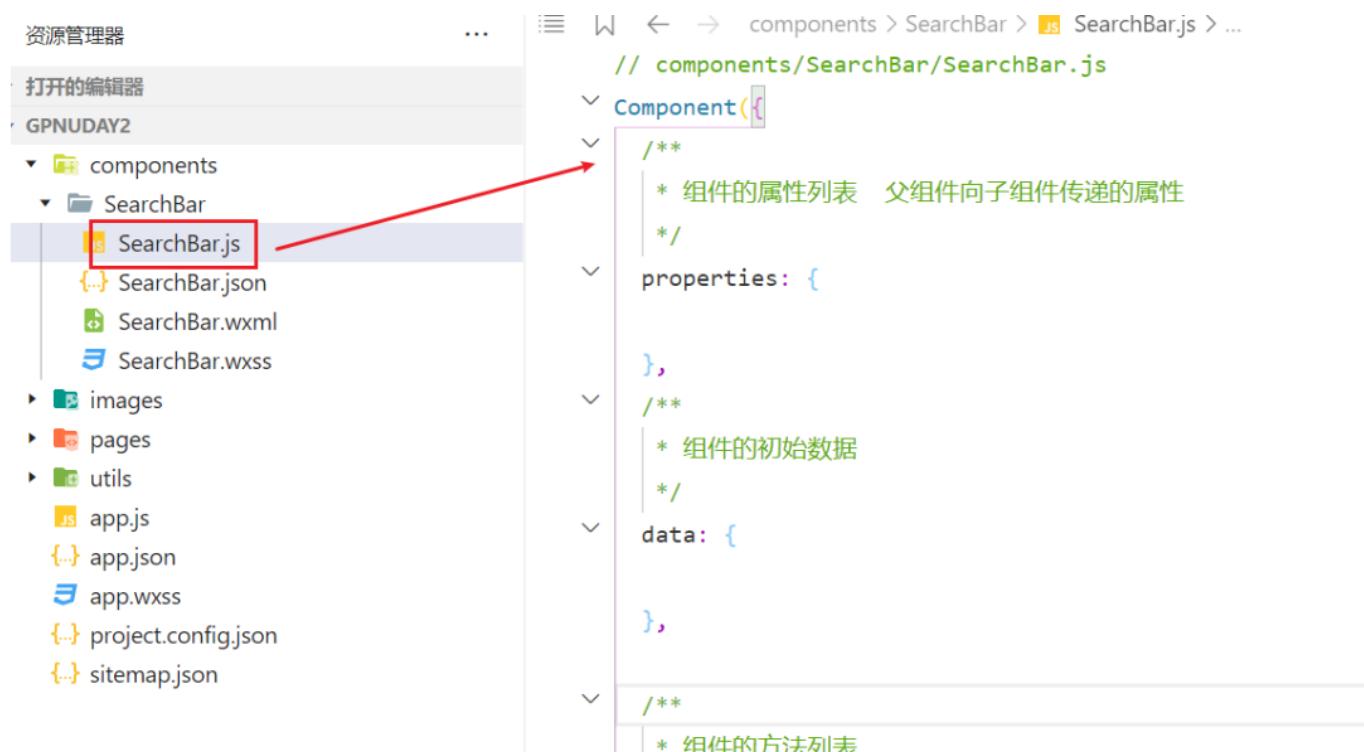
2.配置组件json 文件

这里配置Component 构造器 代表 开启构造器

一个自定义组件也可以视为一个页面，故页面也可以使用 Component 构造器构造，拥有与普通组件一样的定义字段与实例方法，其必要配置项（ json ）与正常自定义组件一致，即需要有 component: true 字段。



3. 配置组件js文件



4. 在SearchInput.wxml 文件中写入内容

```
<view class="search_input">  
  <navigator url="/pages/logs/logs" open-type="navigate"> 搜索 </navigator>  
</view>
```

这里跳转logs页注意情况，如果在我们的tabBar（app.json）配置了按钮，使用navigator跳转不生效，建议换个地址跳转

5. 在SearchInput.wxss 配置样式

```
.search_input {  
    height: 90rpx;  
    padding: 10rpx;  
    background-color: var(--themeColor);  
}  
  
.search_input navigator {  
    height: 100%;  
    display: flex;  
    justify-content: center;  
    align-items: center;  
    background-color: #fff;  
    border-radius: 15rpx;  
    color: #666;  
}
```

app.wxss

```
page{  
    /*定义主题颜色*/  
    --themeColor:#eb4450;  
}
```

6. 引入自定义组件

以在 /pages/index/index 父组件中引入子组件为例

a. 在父组件json 文件的usingComponents 中导入组件



b. 在父组件wxml 文件中以组件名称作为标签使用组件

The screenshot shows the WeChat Mini Program development interface. On the left is the Resource Manager with files like index.json, index.wxml, and index.js. The main area is the WXML editor with the code:

```
<!-- 使用子定义组件 -->
<SearchBar></SearchBar>
```

The code is highlighted with a red box. Below the editor is a toolbar with icons for simulator, editor, debugger, visualization, and cloud development. The preview window at the bottom shows a mobile phone screen with the text "微信小程序day2" and a search bar component.

7. 组件参数传递 - 父传子

注：与VUE父子组件传参原理一样

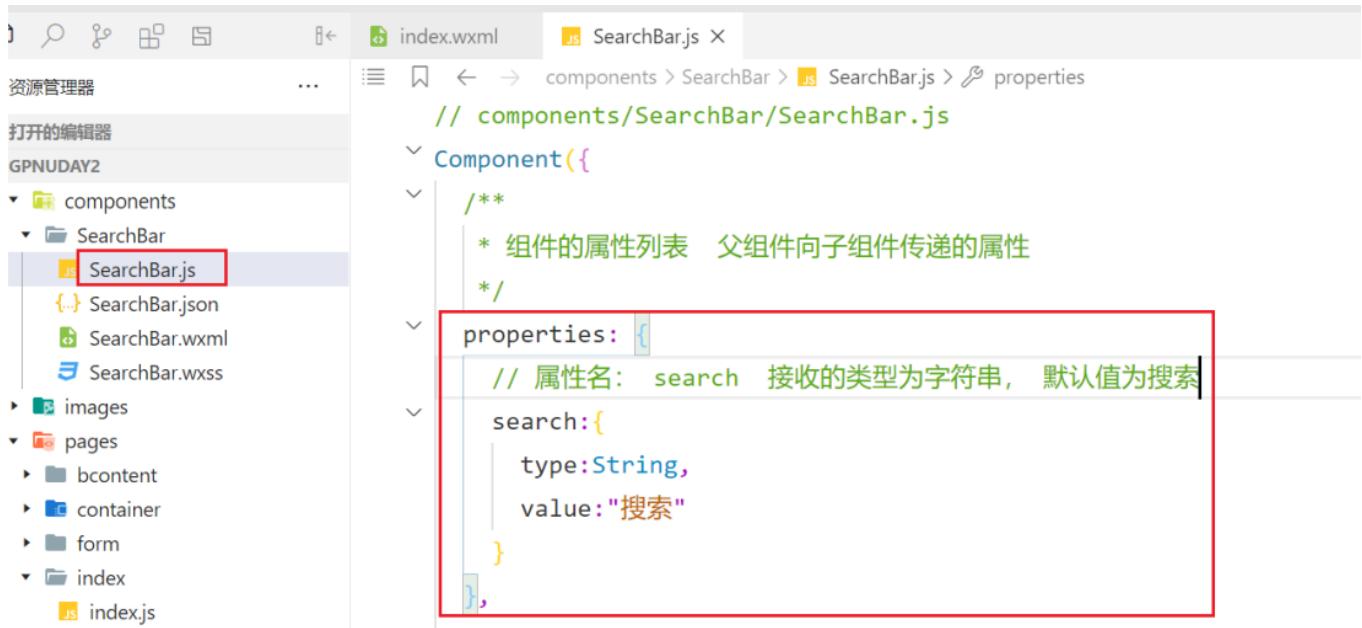
在父组件中的子组件标签添加属性，给子组件传递数据

The screenshot shows the WeChat Mini Program development interface. The Resource Manager on the left includes index.wxml, index.wxss, and nav.js. The WXML editor on the right contains the following code:

```
<!--index.wxml-->
<!-- 自定义组件 -->
<SearchBar search="look"></SearchBar>
```

The "search" attribute is highlighted with a red box. A note below the editor states: "在父组件中向子组件通过属性传参，属性名为search".

在子组件 js 文件中通过 properties 接收，可以指定接收数据类型



```
// components/SearchBar/SearchBar.js
Component({
  /**
   * 组件的属性列表 父组件向子组件传递的属性
   */
  properties: {
    // 属性名: search 接收的类型为字符串, 默认值为搜索
    search: {
      type: String,
      value: "搜索"
    }
  }
})
```

在子组件 wxml 文件中的内容替换为变量



```
<!--components/SearchBar/SearchBar.wxml-->
<view>
  <navigator url="/pages/form/form" open-type="navigate">{{search}}</navigator>
</view>
```

效果如下：



8.父传子进阶

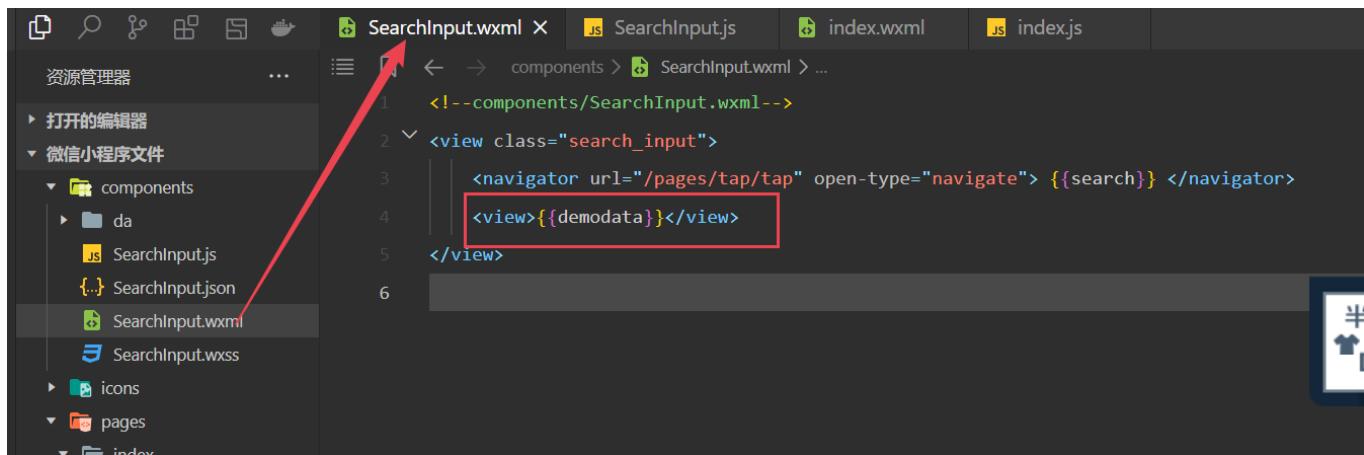
子传父：https://blog.csdn.net/XH_jing/article/details/111563278

1.子组件定义接收变量demodata



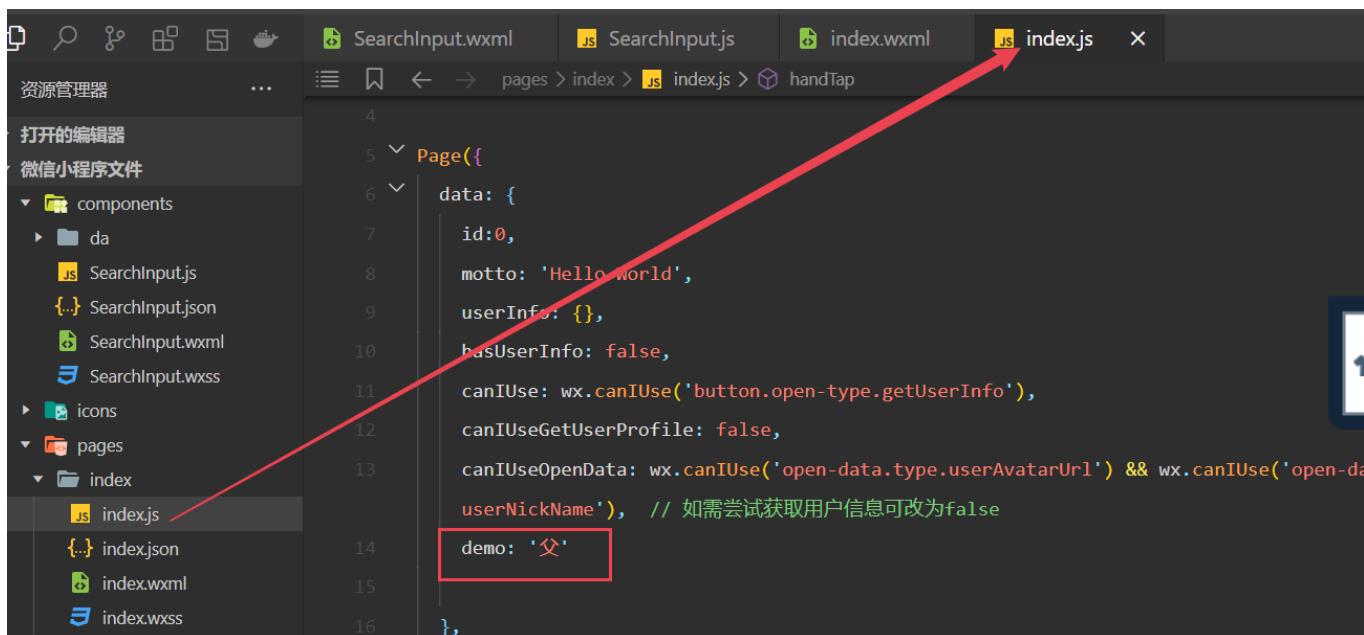
```
// components/SearchInput.js
Component({
  /**
   * 组件的属性列表
   */
  properties: {
    search: {
      type: String,
      value: "搜索自定义组件"
    },
    demodata: String
  },
})
```

2.子组件渲染



```
<!--components/SearchInput.wxml-->
<view class="search_input">
  <navigator url="/pages/tap/tap" open-type="navigate"> {{search}} </navigator>
  <view>{{demodata}}</view>
</view>
```

3.父组件定义操作变量



```
Page({
  data: {
    id: 0,
    motto: 'Hello world',
    userInfo: {},
    hasUserInfo: false,
    canIUse: wx.canIUse('button.open-type.getUserInfo'),
    canIUseGetUserProfile: false,
    canIUseOpenData: wx.canIUse('open-data.type.userAvatarUrl') && wx.canIUse('open-data.type.userNickName'), // 如需尝试获取用户信息可改为false
    demo: '父'
  }
})
```

资源管理器

打开的编辑器

微信小程序文件

- components
- da
- SearchInput.js
- SearchInput.json
- SearchInput.wxml
- SearchInput.wxss
- icons
- pages
- index
- index.js
- index.json
- index.wxml
- index.wxss

SearchInput.wxml

SearchInput.js

index.wxml

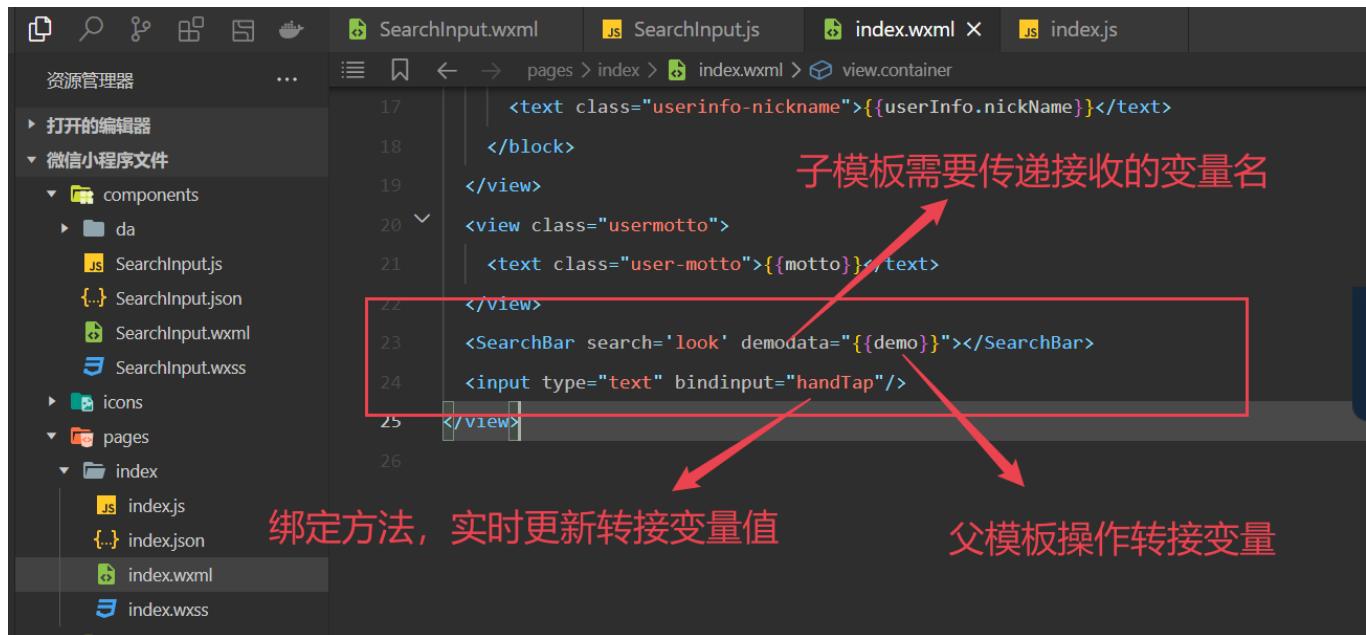
index.js

```
17 |     <text class="userinfo-nickname">{{userInfo.nickName}}</text>
18 |
19 |     </block>
20 |     </view>
21 |     <view class="usermotto">
22 |         <text class="user-motto">{{motto}}</text>
23 |     </view>
24 |     <SearchBar search='look' demodata="{{demo}}"></SearchBar>
25 |     <input type="text" bindinput="handTap"/>
26 | </view>
```

子模板需要传递接收的变量名

绑定方法，实时更新转接变量值

父模板操作转接变量



资源管理器

打开的编辑器

微信小程序文件

- components
- da
- SearchInput.js
- SearchInput.json
- SearchInput.wxml
- SearchInput.wxss
- icons
- pages
- index
- index.js
- index.json
- index.wxml
- index.wxss
- logs

SearchInput.wxml

SearchInput.js

index.wxml

index.js

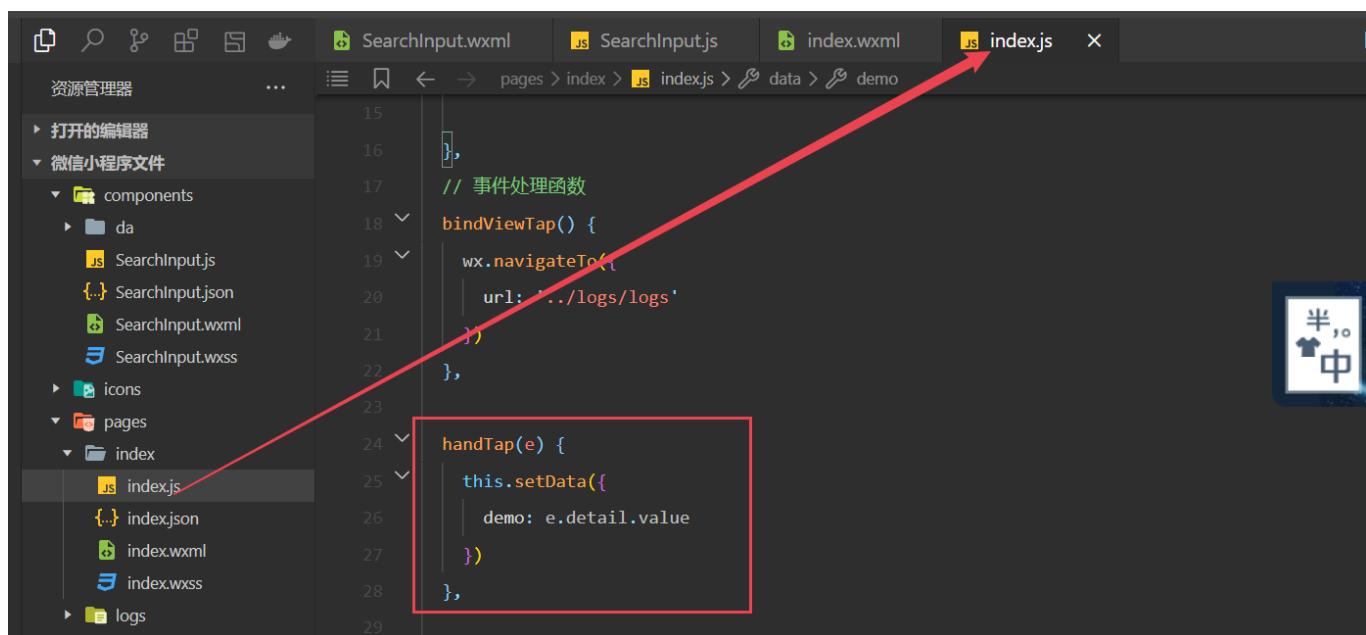
index.json

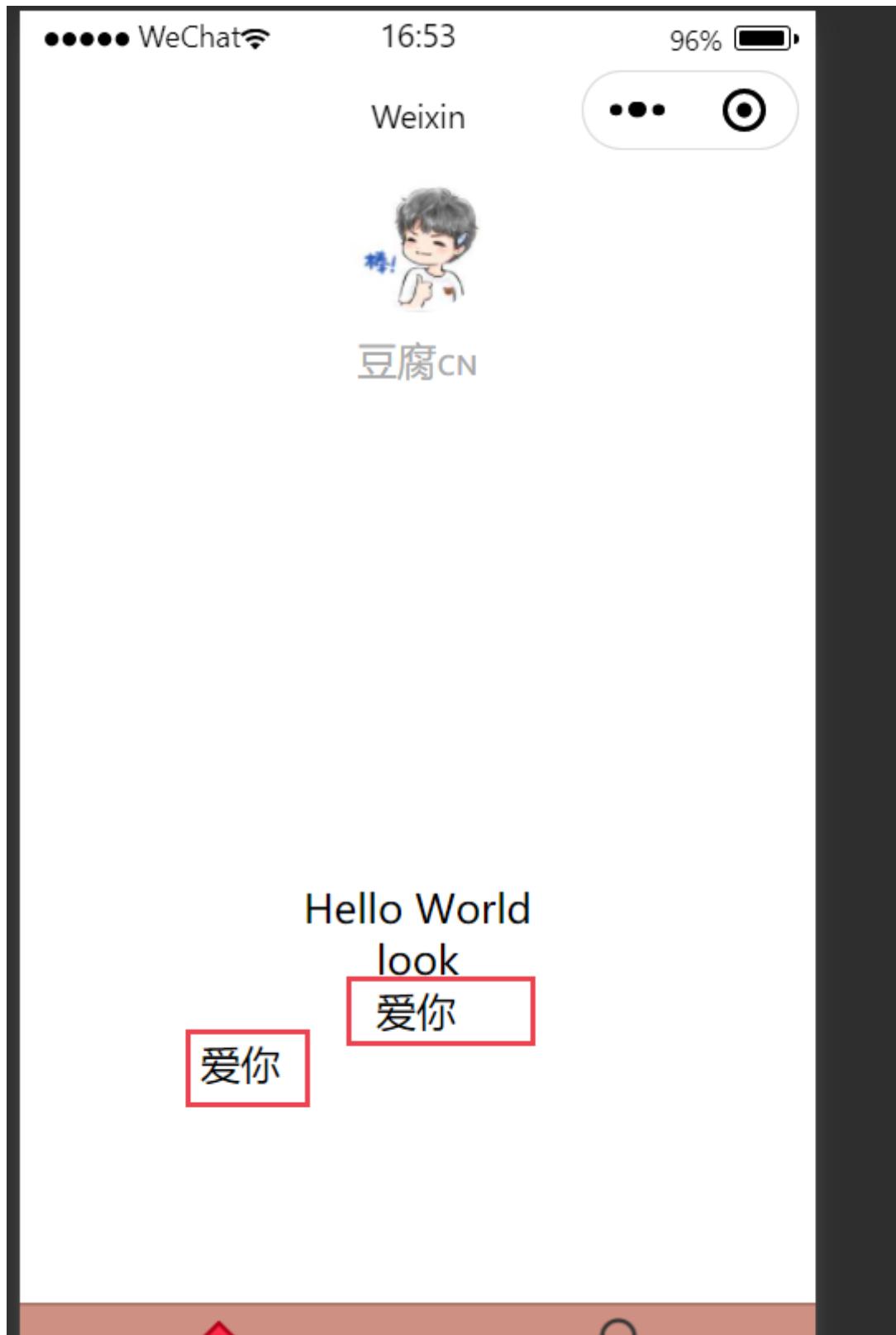
index.wxml

index.wxss

index.js

```
15 |
16 |
17 | },
18 | // 事件处理函数
19 | bindViewTap() {
20 |     wx.navigateTo({
21 |         url: '../logs/logs'
22 |     });
23 |
24 |     handTap(e) {
25 |         this.setData({
26 |             demo: e.detail.value
27 |         });
28 |     };
29 | }
```





9.子传父

首先在子组件js中建立一个方法，在出触发后调用父组件方法 传递子组件数据

SearchInput.js (components/SearchInput.js)

```
19 },
20 },
21 /**
22 * 组件的方法列表
23 */
24 methods: {
25     handTap(e) {
26         // triggerEvent是立即执行触发事件的脚本 (要触发的事件, 传递的值)
27         // 触发父组件中定义的 myevent
28         this.triggerEvent('myevent', e.detail.value)
29     }
30 },
31 }
```

子组件建立输入框，同时绑定input事件（输入时触发）

SearchInput.wxml (components/SearchInput.wxml)

```
<!--components/SearchInput.wxml-->
<view class="search_input">
    <navigator url="/pages/tap/tap" open-type="navigate"> {{search}} </navigator>
    输入: <input type="text" bindinput="handTap"/>
</view>
```

父组件建立接收变量

index.js (pages/index/index.js)

```
4
5 Page({
6     data: {
7         id: 0,
8         motto: 'Hello World',
9         userInfo: {},
10        hasUserInfo: false,
11        canIUse: wx.canIUse('button.open-type.getUserInfo'),
12        canIUseGetUserProfile: false,
13        canIUseOpenData: wx.canIUse('open-data.type.userAvatarUrl') && wx.canIUse('open-data.type.
14        userNickName'), // 如需尝试获取用户信息可改为false
15        demo: ''
16    },
17 })
```

父组件建立方法事件-将子组件调用此方法传递过来的数据处理

The screenshot shows the WeChat Mini Program development environment. The left sidebar displays the project structure under '微信小程序文件': components (SearchInput), icons, pages (index). The right side shows the code editor for 'index.js'.

```
18 bindViewTap() {  
19     wx.navigateTo({  
20         url: '../logs/logs'  
21     })  
22 },  
23  
24 myevent: function (e) {  
25     console.log(e)  
    // 传递过来的值会放在e对象的detail里面  
26     this.setData({  
27         demo: e.detail  
28     })  
29 },  
30  
31 },
```

父组件渲染展示

The screenshot shows the WeChat Mini Program development environment. The left sidebar displays the project structure under '微信小程序文件': components (SearchInput), icons, pages (index). The right side shows the code editor for 'index.wxml'.

```
17 <text class="userinfo-nickname">{{userInfo.nickName}}</text>  
18 </block>  
19 </view>  
20 <view class="usermotto">  
21     <text class="user-motto">{{motto}}</text>  
22 </view>  
23 <!-- bindmyevent 将 myevent 方法与 子组件绑定 -->  
24 <SearchBar search='look' bindmyevent="myevent"></SearchBar>  
25     index展示: <view>{{demo}} - 1</view>  
26 </view>
```