小程序开发文档02/27

# 一、自定义组件

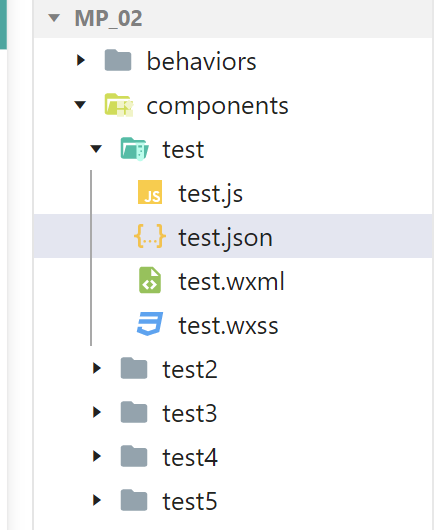
## 1、创建组件

1）在项目的根目录中右键创建components->test文件夹

2）在新建的test文件夹中，右键点击新建component

3）键入组件的名称后回车，会自动生成组件对应的4个文件，后缀名分别为.js .json .wxml和.wxss

注意：为了保证目录结构清晰，建议把不同的组件，存放到单独目录中。



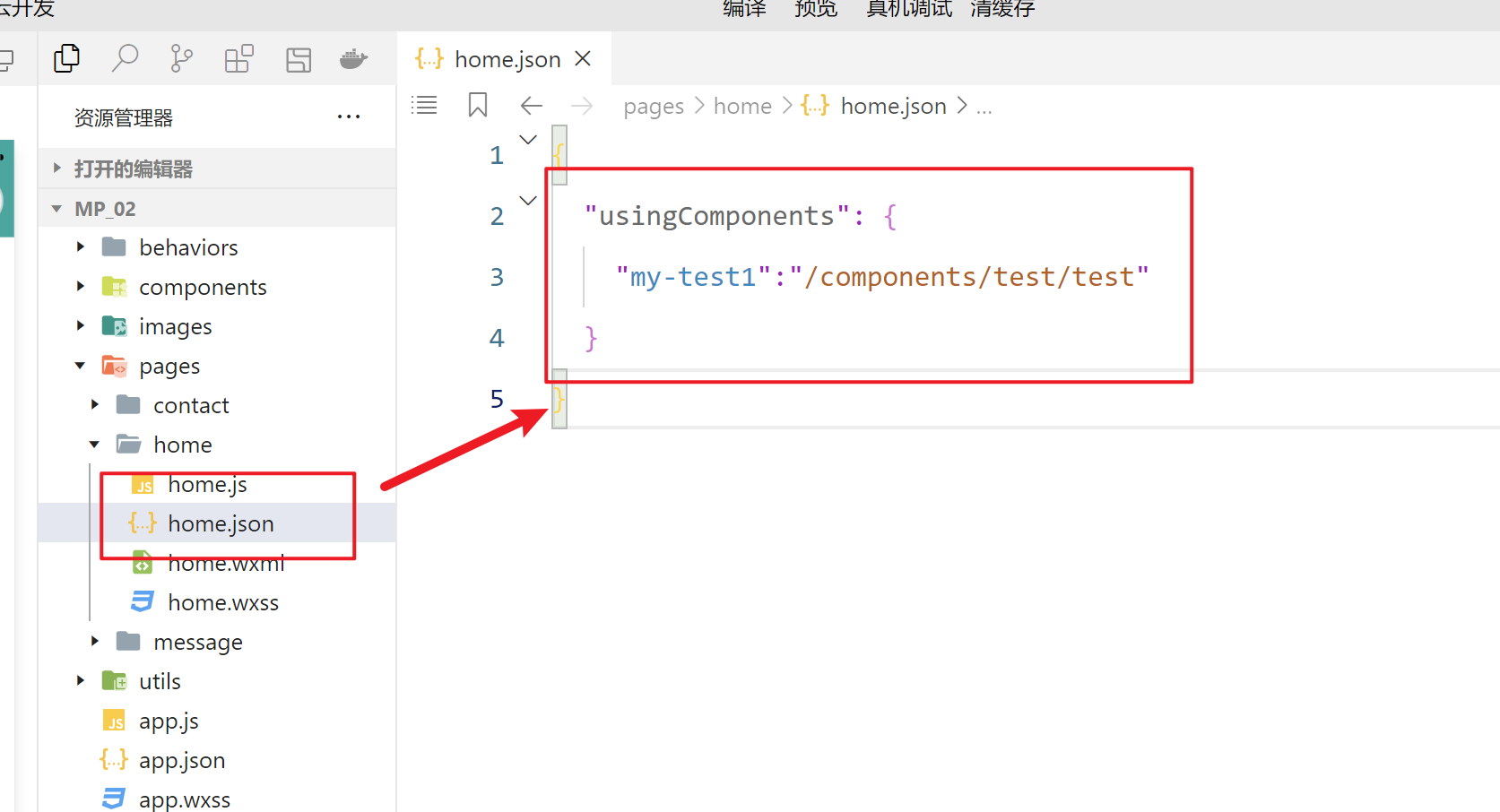
## 2、引用组件

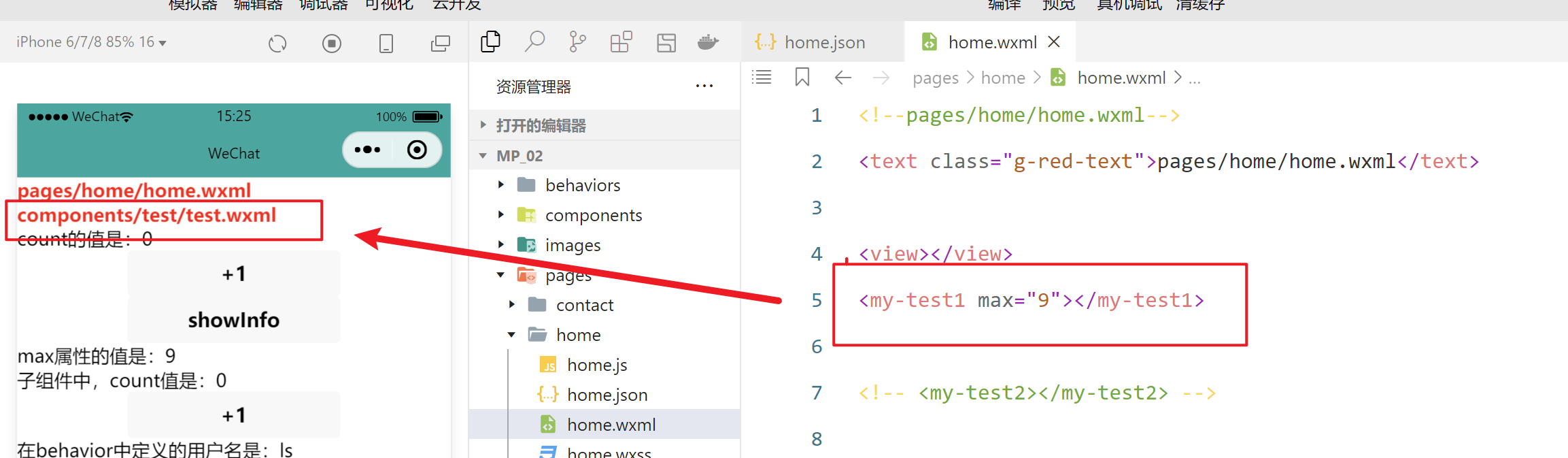
局部引用：组件只能在当前被引用的页面内使用

全局引用：组件可以在每个小程序页面中使用

## 3、局部引用

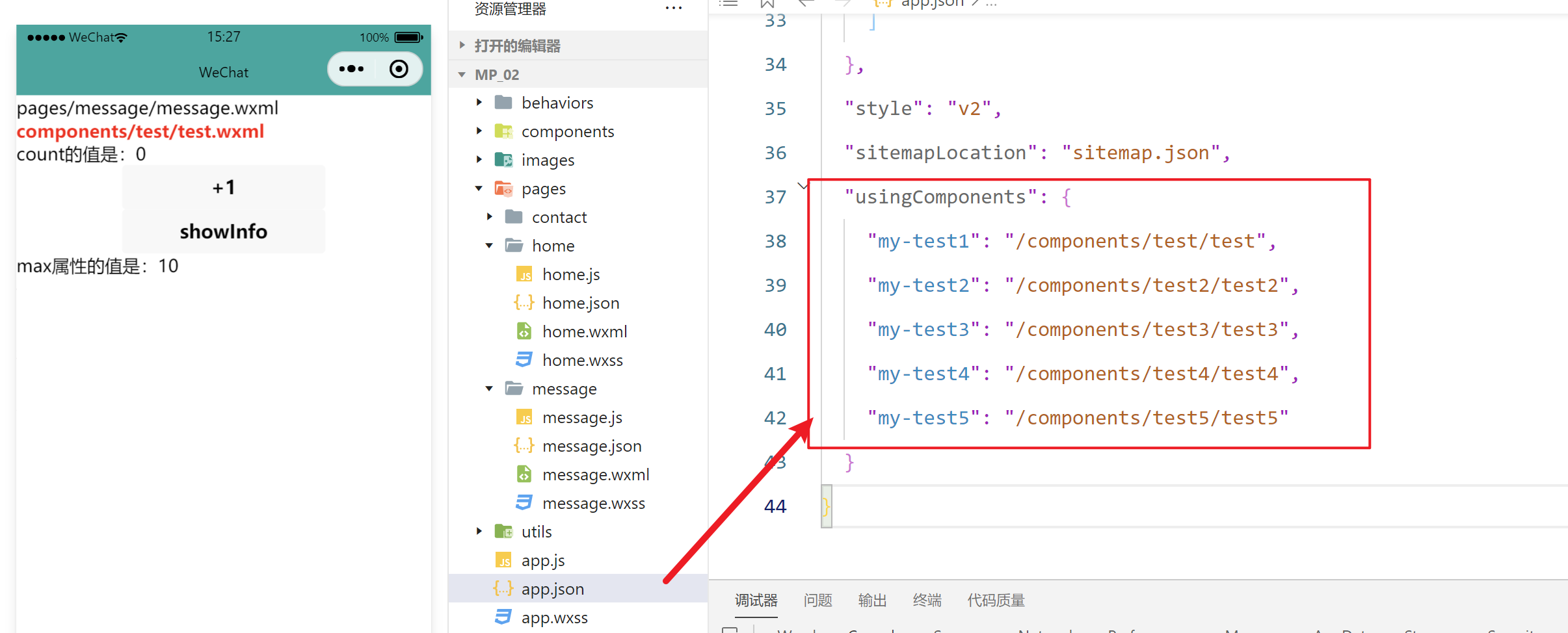
在页面.json配置文件中引用组件的方式，加做“局部应用“





## 4、全局引用组件

在app.json全局配置中引用。



## 5、全局引用和局部引用

根据组件的使用频率和范围，选择不同的引用方式。

## 6、组件和页面的区别

表面上看，文件结构组成是一样的。但组件和页面的.js与.json文件是有明显不同：

组件的.json文件中需要声明”Component”:ture属性；

组件的.js文件中调用的是Component()函数；

组件的事件处理函数需要定义到menthods节点中。





# 二、组件样式隔离

## 1、隔离特性

1）默认情况下，自定义组件的样式只对当前组件生效，不会影响组件以外的ui结构

2）只有class选择器有样式隔离效果。

## 2、修改组件的样式隔离选项

默认情况下，自定义组件的样式隔离特性能够防止组件内样式互相干扰的问题。但可以通过stylesolation修改组件的样式隔离选项







# 三、自定义组件-数据、方法和属性

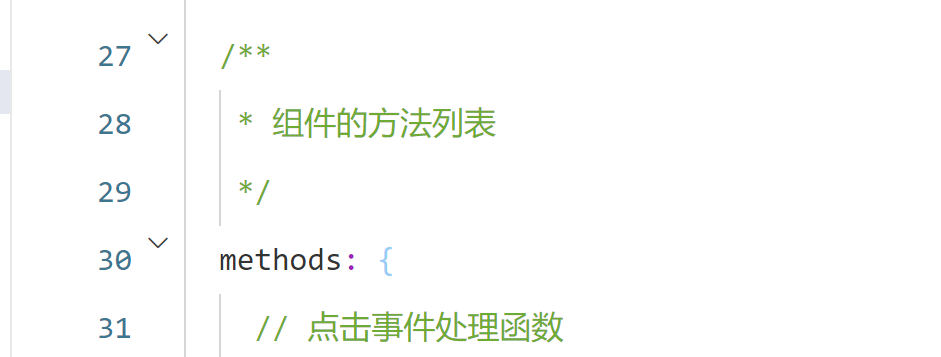
## 1、data数据

在小程序组件中，用于组件模板渲染的私有数据，需要定义到data节点中



## 2、methods方法

在小程序组件中，事件处理函数和自定义方法，需要定义到methods节点中



## 3、properties属性

在小程序组件中，properties是组件的对外属性，用来接收外界传递到组件中的数据



## 4、data和properties的区别

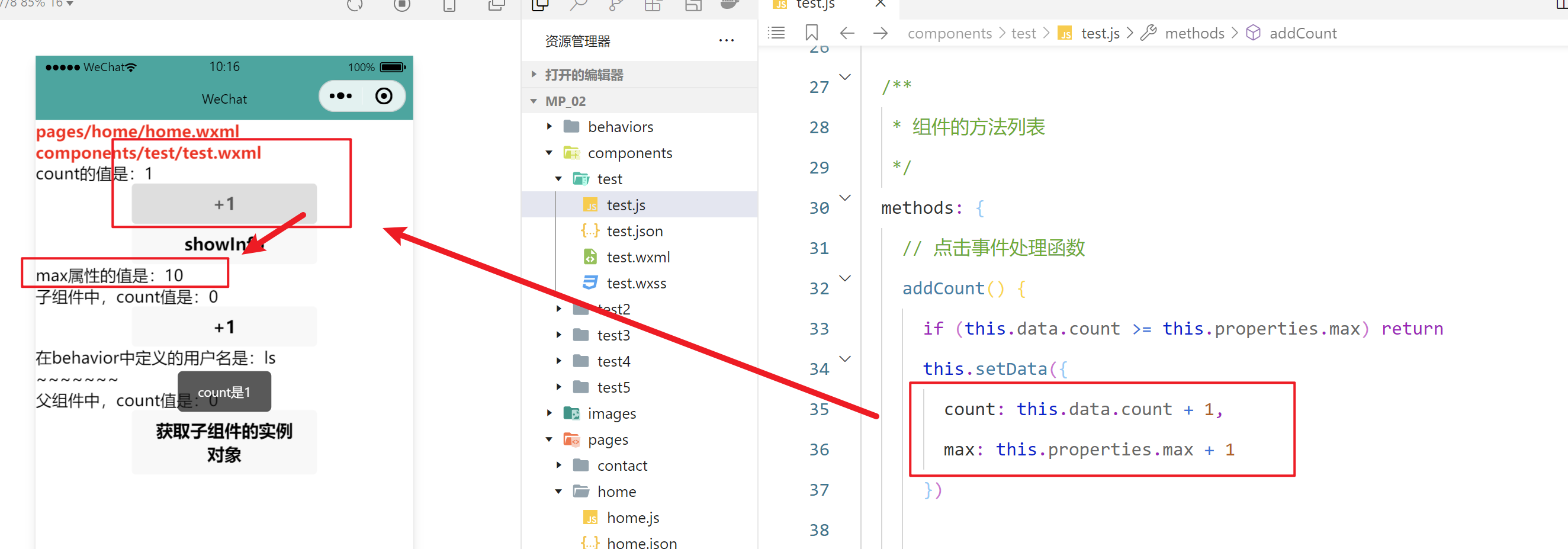
Data和properties用法相同，都是可读可写，但：

Data更倾向于存储组件的私有数据

Properties更倾向于存储外界传递到组件中的数据

## 5、使用setData修改properties的值

由于data值和properties属性在本质是一致的，因此properties属性的值也可以用于页面渲染，或使用setData为properties中的属性重新赋值



# 四、自定义组件-数据监听器

## 1、介绍

数据监听器用于监听和响应任何属性或数据字段的变化，从而执行特定的操作。它作用类型于vue中的watch侦听器。

## 2、数据监听器的基本用法



## 3、监听对象属性的变化

数据监听器支持监听对象中的单个或多个属性的变化

# 五、自定义组件-纯数据字段

## 1、介绍

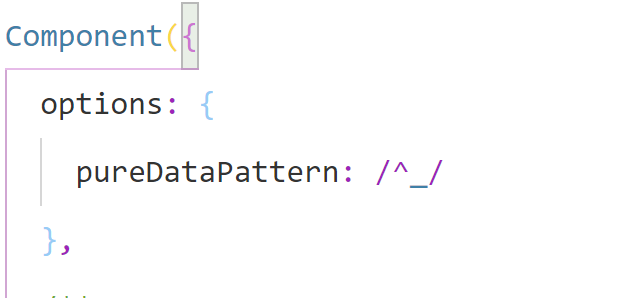
指不用于界面渲染的data字段

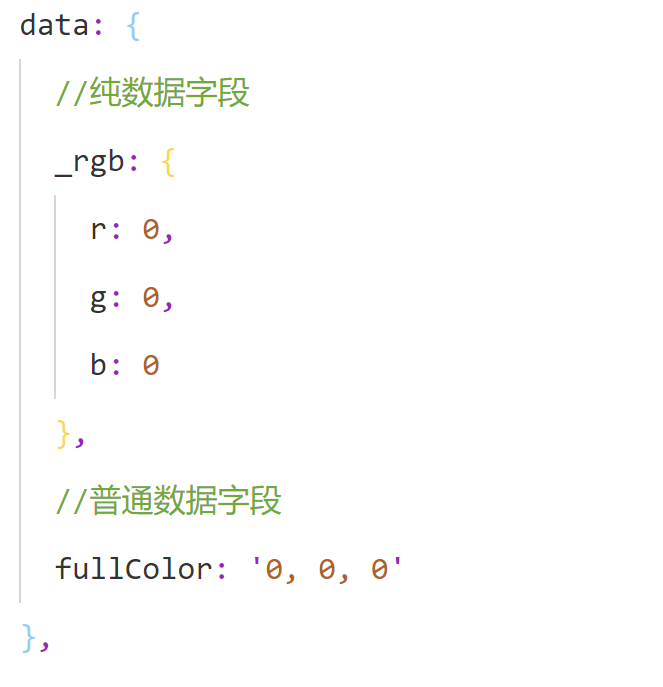
应用场景：某些data中的字段既不会展示在界面上，也不会传递给其他组件，仅仅在当前组件内部使用。带有这种特性的data字段适合被设置为纯数据字段。

好处：有助于提升页面更新的性能。

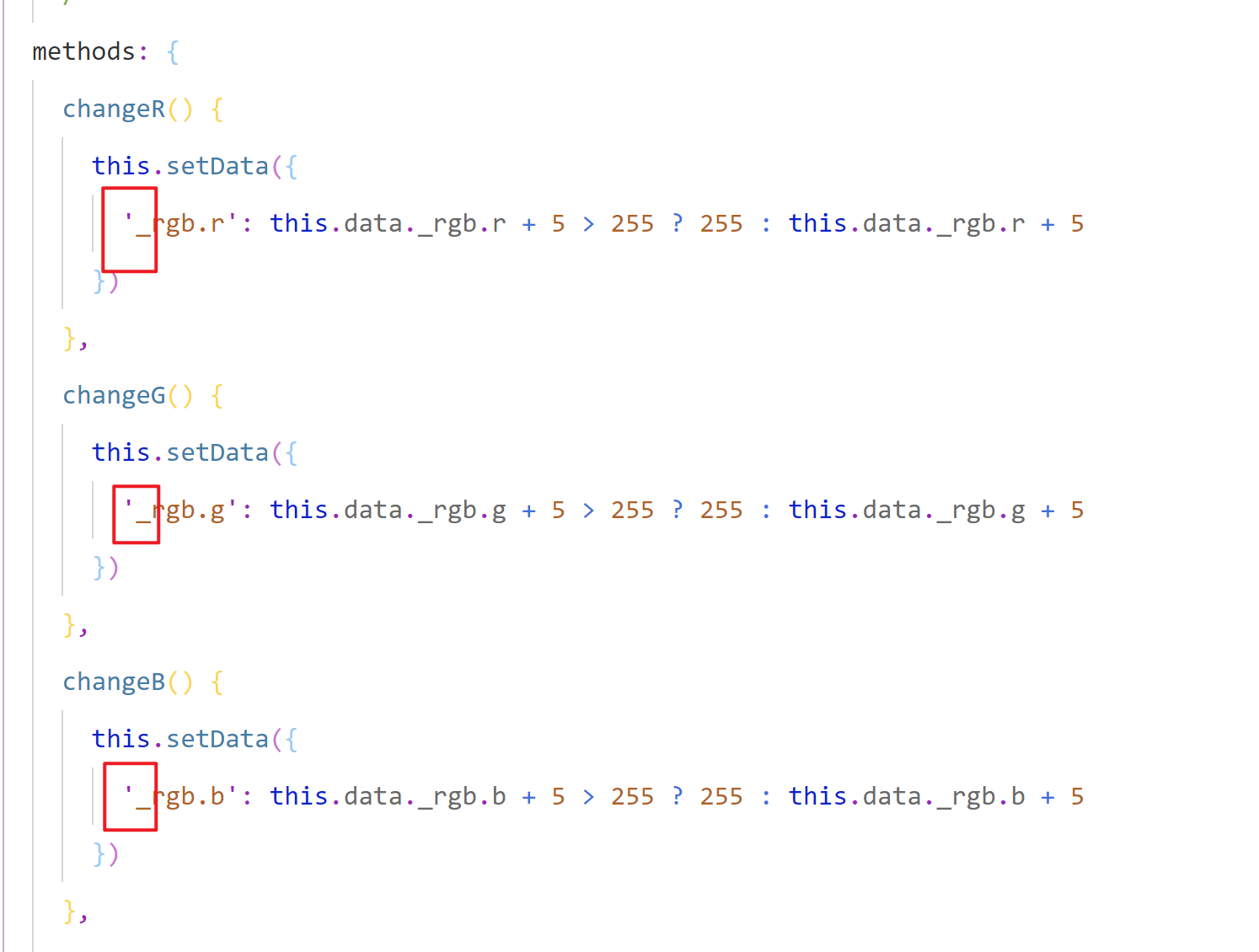
## 2、使用规则

在Component构造器的option节点中，指定pureDataPattern为一个正则表达式，字段名符合这个正则表达式将成为纯数据字段





## 3、使用纯数据字段改造监听器



# 六、组件的生命周期

## 1、组件全部的生命周期函数

|  |  |  |
| --- | --- | --- |
| 生命周期函数 | 参数 | 描述说明 |
| created | 无 | 在组件实例刚被创建时执行 |
| attached | 无 | 在组件实例进入页面节点树时执行 |
| ready | 无 | 在组件在视图层布局完成后执行 |
| moved | 无 | 在组件实例被移动到节点树另一个位置时执行 |
| detached | 无 | 在组件实例被从页面节点树移除时执行 |
| error | Object Error | 每当组件方法抛出错误时执行 |

## 2、组件主要生命周期函数

created、attached、detached。特点如下：

1）组件实例刚被创建好时候，created生命周期会被触发

此时不能调用setData

通常在这个生命周期函数中，只应该用于给组件的this添加一些自定义的属性字段

2）组件完成初始化完毕、进入页面节点树后。attached会被触发

此时，this.data已被初始化完毕

绝大多数初始化工作可以在这个时机进行（例如发请求获取初始数据）

3）在组件离开页面节点树后，detached生命周期函数会被触发

退出一个页面时，会触发页面内每个自定义组件的detached函数

此时适合做清理性质工作

## 3、lifetimes节点

在小程序组件中，生命周期函数可以直接定义在Component构造器的第一季参数中，可以在lifetimes字段内进行声明（推荐的方式，优先级最高）.



# 七、组件所在页面的生命周期

## 1、介绍

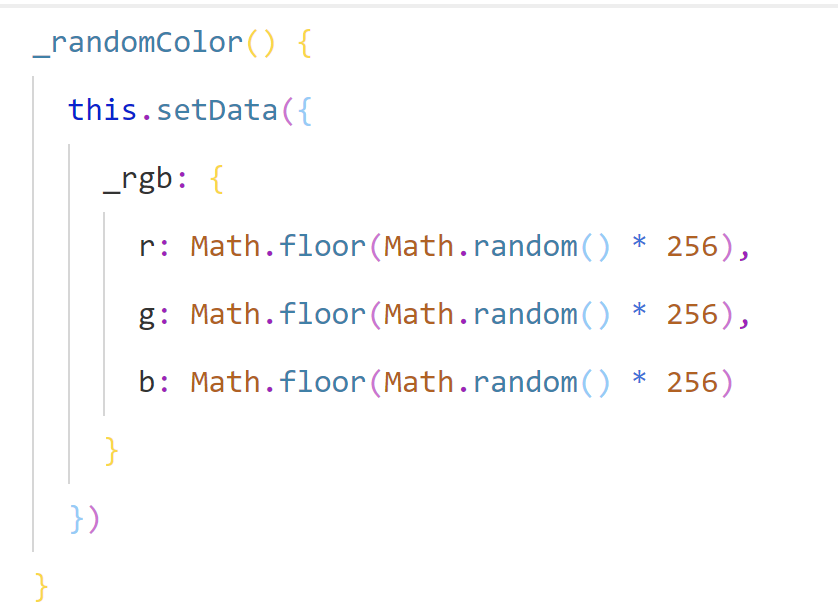
有时自定义组件的行为依赖于页面状态的变化，此时就需要用到组件所在页面的生命周期。

## 2、pageLifetimes节点

组件所在页面的生命周期函数，需要定义在pageLifetime节点中。



## 3、生成随机的RGB颜色值.



# 八、自定义组件-插槽

## 1、介绍

在自定义组件的wxml结构中，可以提供一个<solt>节点（插槽），用于承载组件使用者提供的wxml结构。

## 2、单个插槽

默认每个自定义组件中只允许一个<slot>进行占位，这种个数上的限制叫做单个插槽。





## 3、启用多个插槽

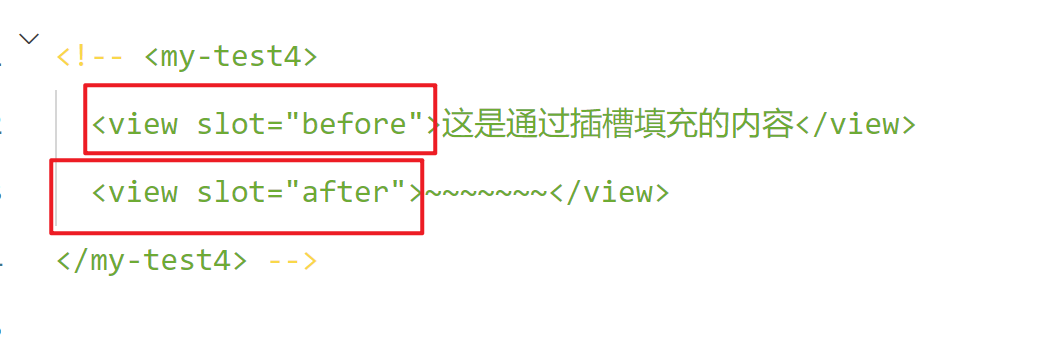
需要时，可以在组件的.js文件中，通过如下方式进行启用：



## 4、定义多个插槽

在组件.wxml中使用多个<slot>标签，以不同的name来区分不同的插槽。





# 九、父子组件之间的通信

## 1、父子组件之间的3中通信方式

1）属性绑定

用于父组件向子组件的指定属性设置数据，仅能设置JSON兼容的数据

2）事件绑定

用于子组件向父组件传递数据，可以传递任意数据

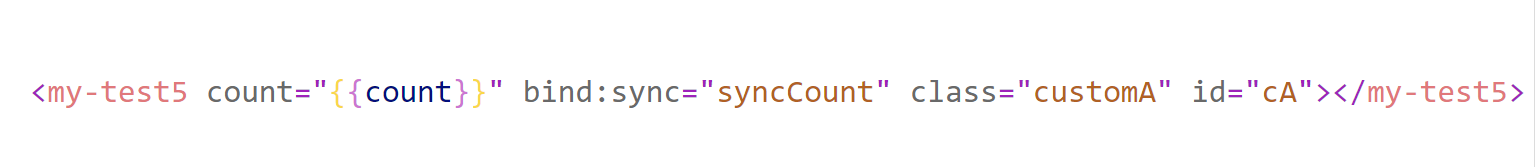
3）获取组件实例

父组件还可以通过this.selectComponent()获取子组件实例对象

这样可以直接访问子组件的任意数据和方法

## 2、属性绑定

用于实现父向子传值，而且只能传递普通类型的数据，无法传递方法给子组件。



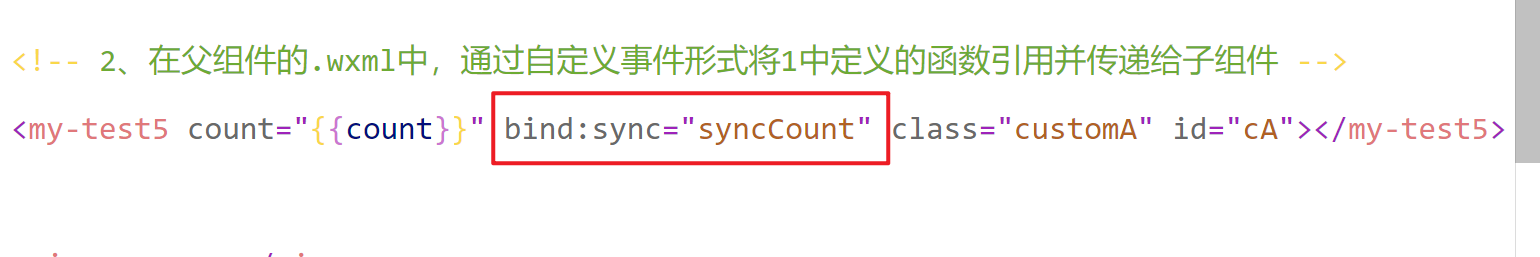
子组件在properties节点中声明对应的属性并使用

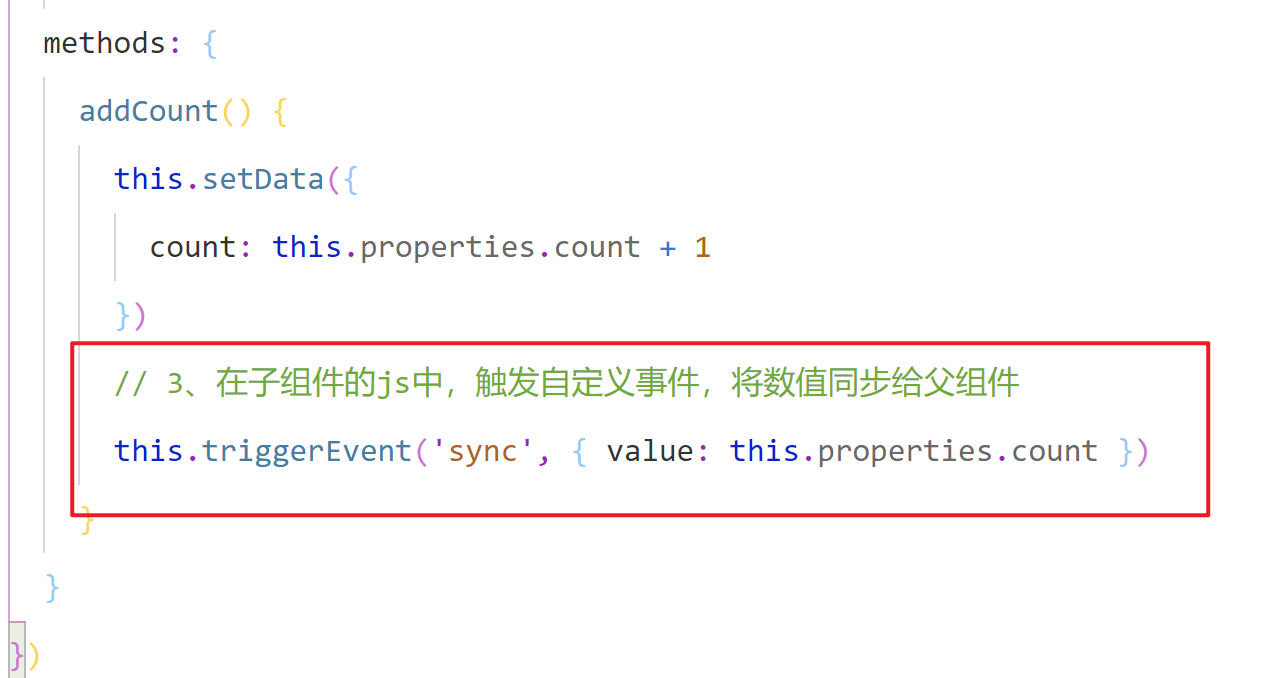


## 3、事件绑定

用于实现子向父传值，可以传递任何类型数据



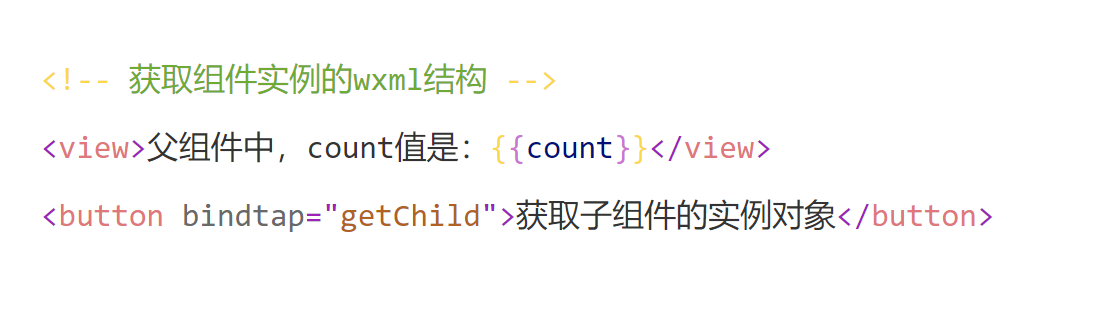






## 4、获取组件实例

可以在父组件里调用this.selectComponent(“id或class选择器”)，获取子组件的实例对象，从而直接访问子组件的任意数据和方法。调用时需要传入一个选择器。





# 十、自定义组件-behaviors

## 1、介绍

Behaviors是小程序中，用于实现组件间代码共享的特性，类似于vue.js中的“mixins”。

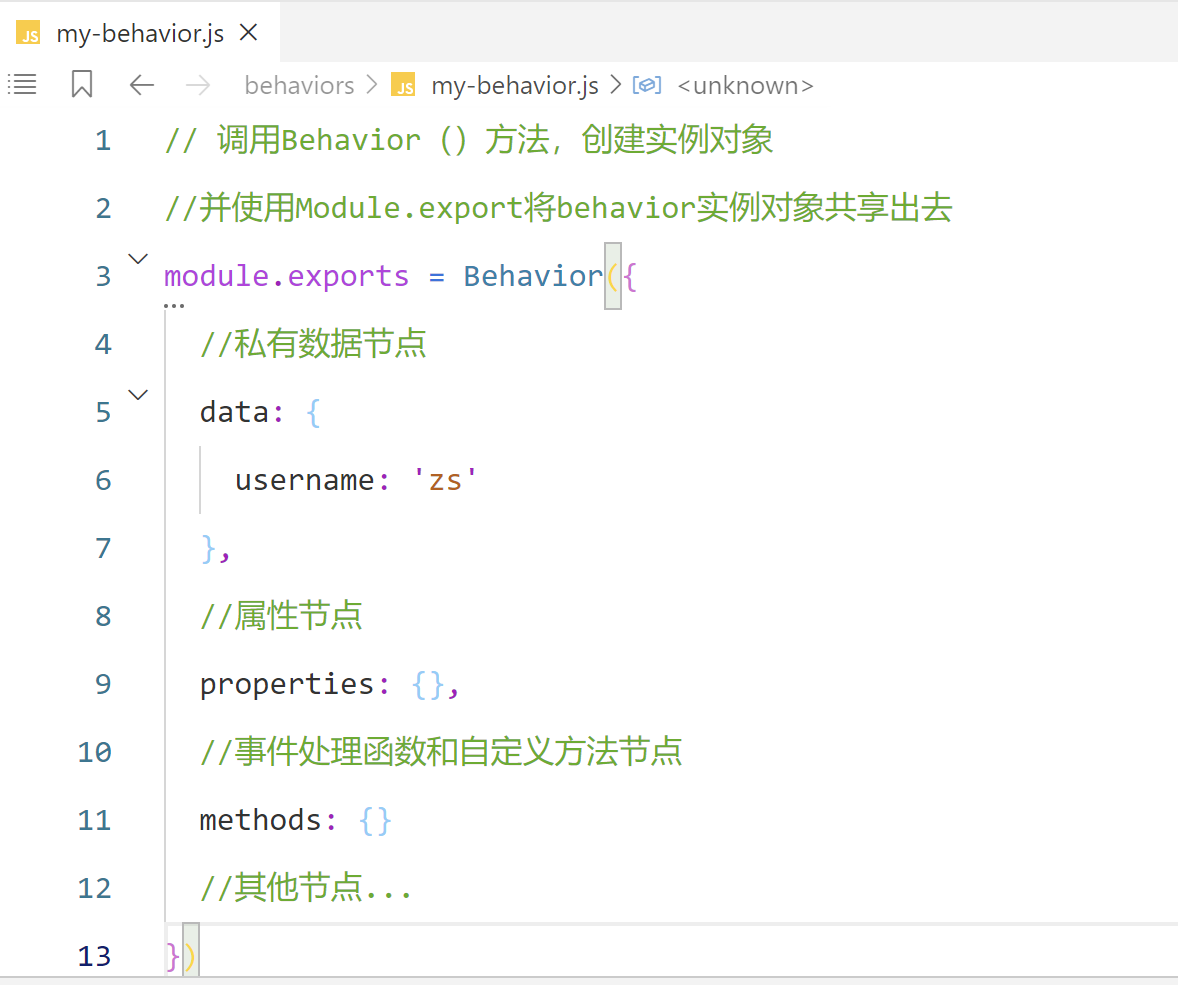
## 2、工作方式

每个behavior可以包含一组属性、数据、生命周期函数和方法。组件引用它时，它的属性、数据和方法会被合并到组件中。

每个组件可以引用多个behavior，behavior也可以引用其他behavior。

## 3、创建behavior

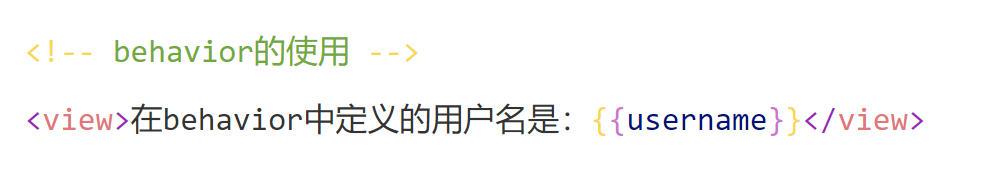
调用Behavior方法即可创建一个共享的Behavior实例对象，供所有的组件使用：



## 4、导入并使用Behavior

在组件中，使用require()方法导入需要的behavior，挂载后即可访问behavior中的数据或方法。





## 5、behavior中主要可用的节点

|  |  |  |  |
| --- | --- | --- | --- |
| 可用节点 | 类型 | 是否必填 | 描述 |
| properties | Object Map | 否 | 同组件的属性 |
| data | Object | 同组件的数据 |
| methods | Object | 同自定义组件方法 |
| behaviors | String Array | 引入其他的behavior |

## 6、同名字段的覆盖和组合规则

1）同名数据字段（data）

2）同名的属性（properites）或方法（methods）

3）同名的生命周期函数

# 十一、总结

1、能够创建并引用组件

2、能够知道如何修改组件的样式隔离选项

3、能够知道如何定义和使用数据监听器

4、能够知道如何定义和使用纯数据字段

5、能够知道实现组件父子通信有娜三种方式

5、能够知道如何定义和使用behaviors