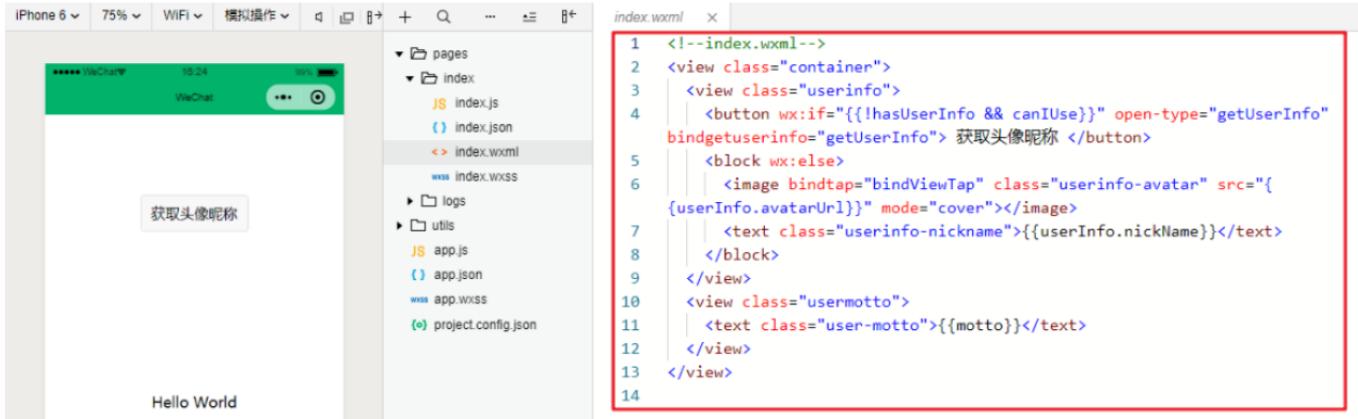


# 1. 概述

官方文档: <https://developers.weixin.qq.com/miniprogram/dev/framework/quickstart/code.html#WXML-%E6%A8%A1%E6%9D%BF>

从事过网页编程的人知道，网页编程采用的是 `HTML + CSS + JS` 这样的组合，其中 `HTML` 是用来描述当前这个页面的结构，`CSS` 用来描述页面的样子，`JS` 通常是用来处理这个页面和用户的交互。

同样道理，在小程序中也有同样的角色，其中 `WXML` 充当的就是类似 `HTML` 的角色。打开 `pages/index/index.wxml`，你会看到以下的内容：



The screenshot shows the WeChat DevTools interface. On the left, there's a preview window for an iPhone 6 showing a simple UI with a button labeled "获取头像昵称". To the right is a file tree and a code editor. The file tree shows a project structure with files like index.js, index.json, index.wxml, index.wxss, logs, utils, app.js, app.json, app.wxss, and project.config.json. The code editor displays the content of index.wxml:

```
index.wxml
1 <!--index.wxml-->
2 <view class="container">
3   <view class="userinfo">
4     <button wx:if="{{!hasUserInfo && canIUse}}" open-type="getUserInfo"
      bindgetuserinfo="getUserInfo" 获取头像昵称 </button>
5     <block wx:else>
6       <image bindtap="bindViewTap" class="userinfo-avatar" src="{{userInfo.avatarUrl}}" mode="cover"></image>
7       <text class="userinfo-nickname">{{userInfo.nickName}}</text>
8     </block>
9   </view>
10  <view class="usermotto">
11    <text class="user-motto">{{motto}}</text>
12  </view>
13 </view>
14
```

和 `HTML` 非常相似，`WXML` 由标签、属性等等构成。但是也有很多不一样的地方，我们来一一阐述一下：

- 1. 标签名字有点不一样
- 2. 多了一些 `wx:if` 这样的属性以及 `{{表达式}}` 的表示方式

注意：微信小程序 借鉴了 `vue.js` 框架的一些优秀理念

# 2. 数据绑定

开发文档: <https://developers.weixin.qq.com/miniprogram/dev/reference/wxml/data.html>

数据绑定使用 `Mustache` 语法（双大括号）将变量包起来

---- 通过 `{{ }}` 的语法把一个变量绑定到界面上，我们称为数据绑定。

---- `WXML` 中的动态数据均来自对应 `js` 文件 `Page` 函数 的 `data` 对象，这样实现了 `wxml` 与 `JavaScript` 之间的数据传递，即数据绑定

小程序提倡把渲染和逻辑分离，简单来说就是不要再让 `js` 直接操控 `DOM`，`js` 只需要管理状态即可，然后再通过一种模板语法来描述状态和界面结构的关系即可。

## ==WXML语法：==

```
<标签>{{变量名}}</标签>
```

## ==常见的数据绑定形式==

### 内容

```
<view> {{ message }} </view>
```

```
Page({
  data: {
    message: 'Hello 双11!'
  }
})
```

### 组件属性(需要在双引号之内)

```
<view id="item-{{id}}"> </view>
```

```
Page({
  data: {
    id: 0
  }
})
```

### 控制属性(需要在双引号之内)

```
<view wx:if="{{condition}}"> 条件判断</view>
```

```
Page({
  data: {
    condition: true
  }
})
```

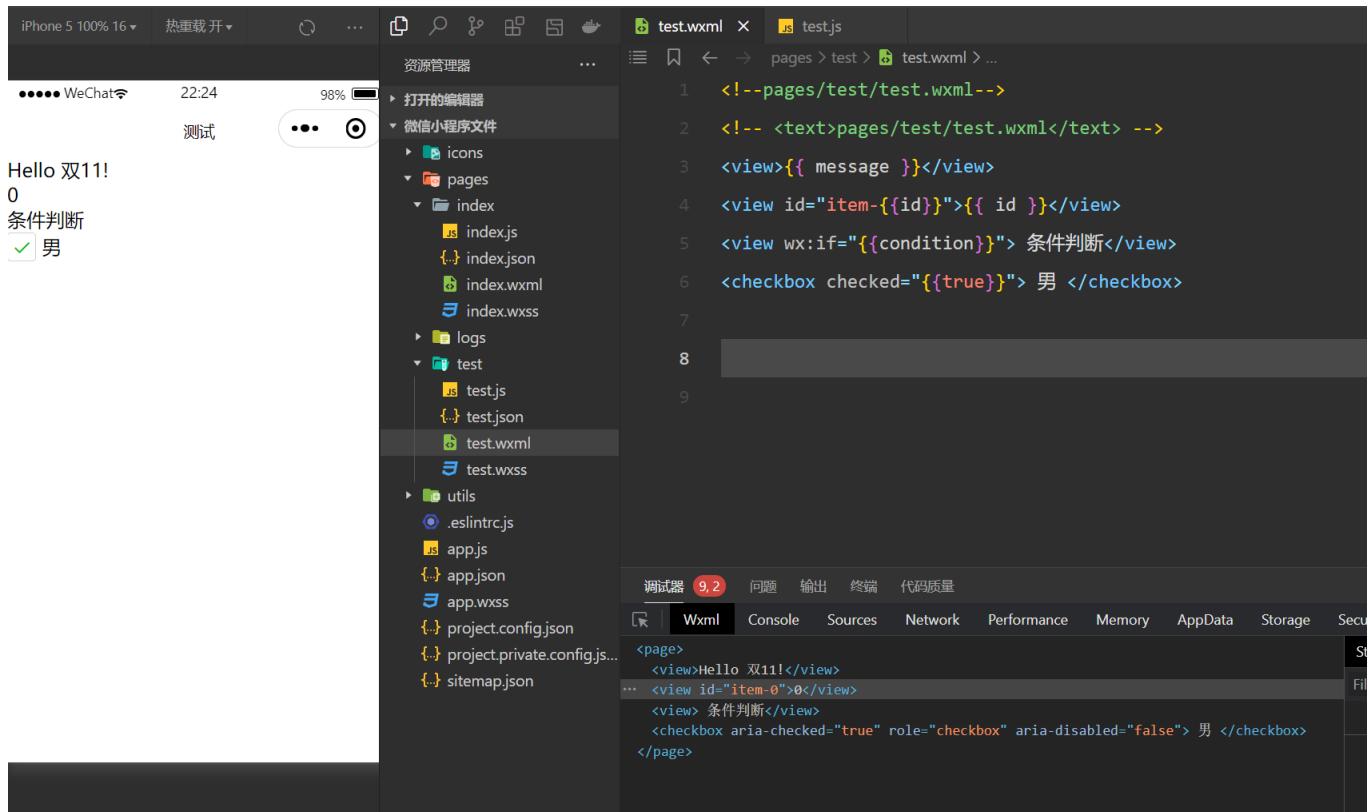
### 关键字(需要在双引号之内)

true : boolean 类型的 true, 代表真值。  
false : boolean 类型的 false, 代表假值。

```
<checkbox checked="{{false}}> 男 </checkbox>
```

特别注意：

不要直接写 checked="false"，  
其计算结果是一个字符串，转成 boolean 类型后代表真值。



## 3. 运算

开发文档: <https://developers.weixin.qq.com/miniprogram/dev/reference/wxml/data.html>  
可以在 {{}} 内进行简单的运算，支持的有如下几种方式：

### ==三元运算==

```
<view hidden="{{num>5 ? true : false}}> Hidden </view>
```

```
data: {
  num: 4,
},
```

hidden: true，则隐藏，为false时显示

? : 问号和冒号是一起用的，叫条件运算符。

语法：

条件表达式 ? 真值 : 假值

这个表达式由三部分组成的，如果条件表达式的值为真，则整个表达式的值为“真值”的值，反之为“假值”的值。

## ==算术运算==

```
<view> {{a + b}} + {{c}} + d </view>

Page({
  data: {
    a: 1,
    b: 2,
    c: 3
  }
})
```

view中的内容为 3 + 3 + d。

## ==逻辑判断==

```
<view wx:if="{{weight > 70}}> 超重</view>

Page({
  data: {
    weight: 80
  }
})
```

## ==字符串运算==

```
<view>{{"hello" + name}}</view>

Page({
  data: {
    name: 'MINA'
  }
})
```

The screenshot shows the WeChat Mini Program development interface. On the left, there's a preview window showing a mobile screen with the text "Hello 双11!" and some calculations like "0", "条件判断", "男", "三元运算", "Hidden", "3 + 3 + d", "超重", and "helloMINA". In the center, there's a file tree with files like test.js, test.wxml, and test.wxss. On the right, there are two code editors for test.js and test.wxml. The test.js code includes a red box around the line `<view>{{"hello" + name}}</view>` and a red box around the variable `name: 'MINA'`. The test.wxml code has a red box around the same line. Below the code editors is a debugger panel with tabs for Wxml, Console, Sources, Network, Performance, Memory, AppData, Storage, Security, Sensor, and a log section showing various app service logs.

==对象==

```
<view>{{student.name}}</view>

Page({
  data: {
    student: {
      name: "张三",
      age: 20,
      gender: "男"
    }
  }
})
```

## 4.列表渲染

开发文档: <https://developers.weixin.qq.com/miniprogram/dev/reference/wxml/list.html>

语法:

```
<标签名 wx:for="{{变量名}}" [wx:for-index="自定义名称" wx:for-item="自定义名称"
wx:key="自定义名称"]>
```

--- wx:for-index 可以指定数组当前下标 的变量名，默认名为: index

--- wx:for-item 可以指定数组当前元素 的变量名（遍历对象的接收变量名），默认名为: item

--- wx:key 可以定义也可以不定义，唯一的标识符 详解--

[https://blog.csdn.net/qq\\_45593068/article/details/118658779](https://blog.csdn.net/qq_45593068/article/details/118658779)

# wx:key 官网解释: <https://developers.weixin.qq.com/miniprogram/dev/reference/wxml/list.html#wx:key>  
# <https://www.cnblogs.com/kenshinobiy/p/9206072.html>

1: `wx:key="字符串"`

这个“字符串”代表在 for 循环的 array 中 item 的某个“属性”  
该“属性”的值需要是列表中唯一的字符串或数字，且不能动态改变。  
用于被遍历的组件需要多个属性的时候。

```
1. //test.js
2. data: {
3.   input_data: [
4.     { id: 1, unique: "unique1" },
5.     { id: 2, unique: "unique2" },
6.     { id: 3, unique: "unique3" },
7.     { id: 4, unique: "unique4" },
8.   ]
9. }
10.
11. //test.wxml
12. <input value="id:{{item.id}}" wx:for="{{input_data}}" wx:key="unique" />
```

2: `wx:key="*this"`

保留关键字“\*this”代表在 for 循环中的 item 本身，  
这种表示需要 item 本身是一个唯一的字符串或者数字  
用于组件仅需要一个属性，且属性值唯一。

```
1. //test.js
2. data: {
3.   numberArray: [1, 2, 3, 4],
4.   stringArray: ['aaa', 'ccc', 'fff', 'good']
5. }
6. //test.wxml
7. <input value="id:{{ item }}" wx:for="{{numberArray}}" wx:key="*this" />
8. <input value="id:{{ item }}" wx:for="{{stringArray}}" wx:key="*this" />
9.
10. ,
```

## ==案例==

首先在 index.js 后缀的文件中设定数组数据

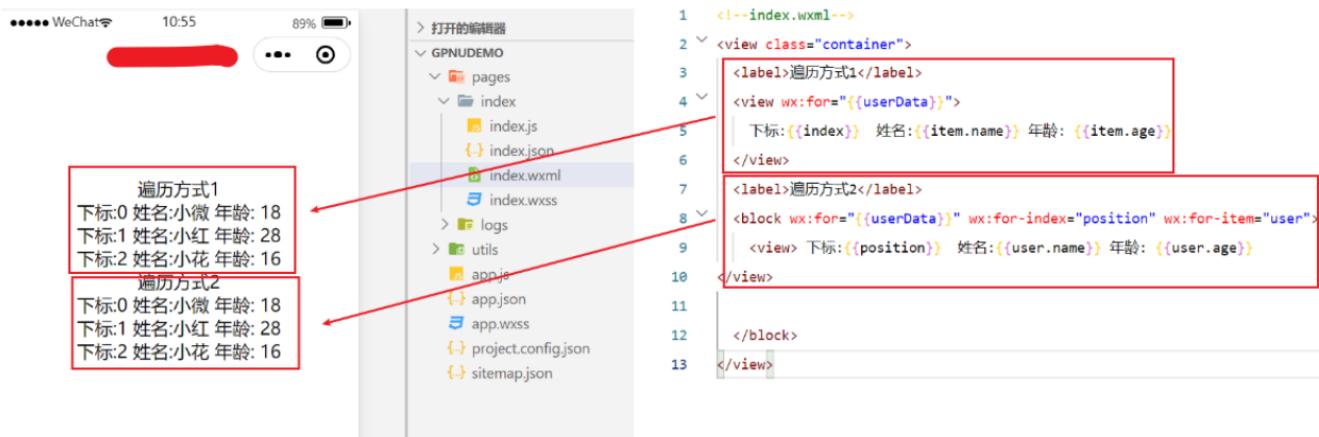
```
Page({
  data: {
    // 定义变量
    userData: [
      {
        name: "小微",
        age: 18,
        height: 169
      },
      {
        name: "小红",
        age: 28,
        height: 165
      },
      {
        name: "小花",
        age: 16,
        height: 155
      }
    ]
  }
})
```

然后在 index.wxml 为后缀的模板文件中使用列表渲染相当于循环遍历出userData的数据

```
<view class="container">
    <label>遍历方式1</label>
    <view wx:for="{{userData}}>
        下标:{{index}} 姓名:{{item.name}} 年龄: {{item.age}}
    </view>

    <label>遍历方式2</label>
    <block wx:for="{{userData}}" wx:for-index="position" wx:for-item="user">
        <view>
            下标:{{position}} 姓名:{{user.name}} 年龄: {{user.age}}
        </view>
    </block>
</view>
```

## 效果



### 注意1:

当 `wx:for` 的值为字符串时，会将字符串解析成字符串数组

```
<view wx:for="array">
    {{item}}
</view>
```

等同于

```
<view wx:for="[['a','r','r','a','y']]>
    {{item}}
</view>
```

### 注意2:

花括号和引号之间如果有空格，将最终被解析成为字符串

```
<view wx:for="[[10,20,30]] " >
    {{item}}
</view>
```

等同于

```
<view wx:for="{{[10,20,30] + ' '}}">
    {{item}}
</view>
```

## 5.条件渲染

开发文档: <https://developers.weixin.qq.com/miniprogram/dev/reference/wxml/conditional.html>

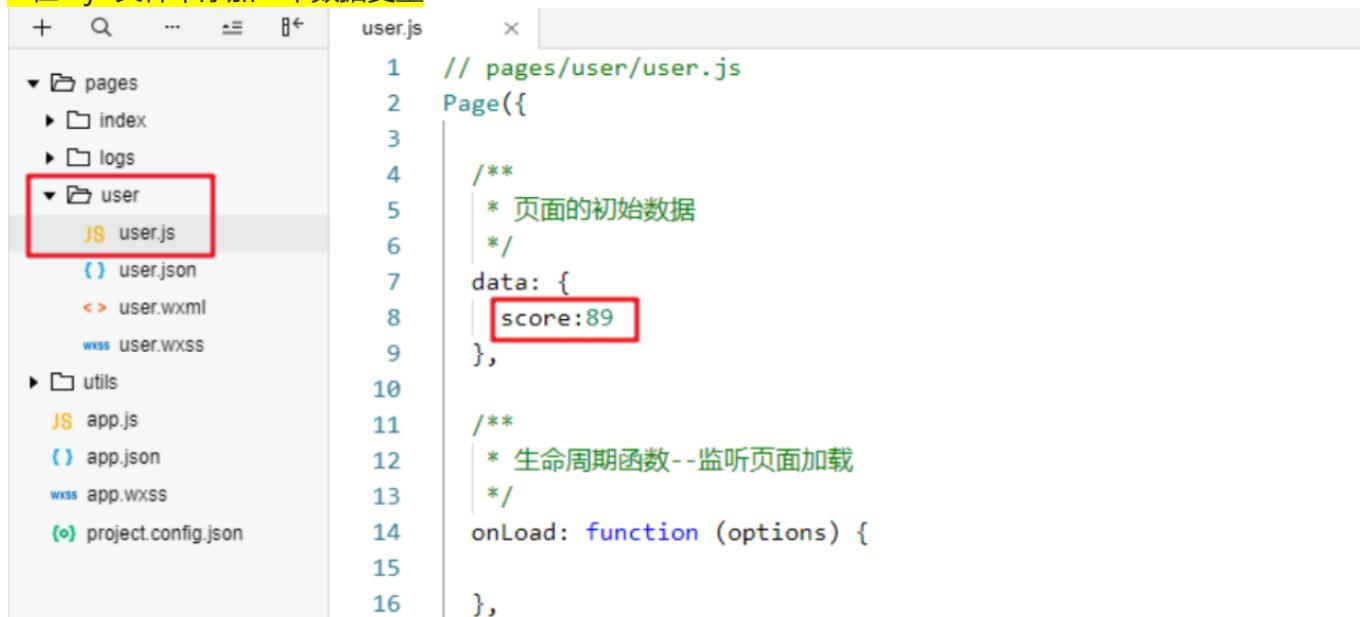
语法:

```
<标签名 wx:if="{{condition}}">
```

在框架中, 使用 `wx:if="{{condition}}"` 来判断是否需要渲染该代码块,  
也可以用 `wx:elif` 和 `wx:else` 来添加一个 `else` 块。

### ==案例==

#### 1.在 \*.js 文件中添加一个数据变量



The screenshot shows a code editor interface with a sidebar on the left displaying a project structure. The 'user' folder is expanded, and its 'user.js' file is selected and highlighted with a red box. The main editor area shows the content of the user.js file:

```
// pages/user/user.js
Page({
  /**
   * 页面的初始数据
   */
  data: {
    score: 89
  },
  /**
   * 生命周期函数--监听页面加载
   */
  onLoad: function (options) {
  }
},
```

#### 2.然后直接在 \*.wxm 模板页面中使用条件渲染

```
<!--pages/users/users.wxml-->
<view>
    <block wx:if="{{score}>=90}">
        <view>分数: {{score}}</view>
        <view>等级: 优秀</view>
    </block>
    <block wx:elif="{{score}>=80}">
        <view>分数: {{score}}</view>
        <view>等级: 良好</view>
    </block>
    <block wx:elif="{{score}>=70}">
        <view>分数: {{score}}</view>
    </block>
```

```
<view>等级: 合格</view>
</block>
<block wx:elif="{{score}>=60}">
    <view>分数: {{score}}</view>
    <view>等级: 及格</view>
</block>
<block wx:else>
    <view>分数: {{score}}</view>
    <view>等级: 不及格</view>
</block>
</view>
```

### 3.效果

