# 微信小程序\_01

# 今日目标8

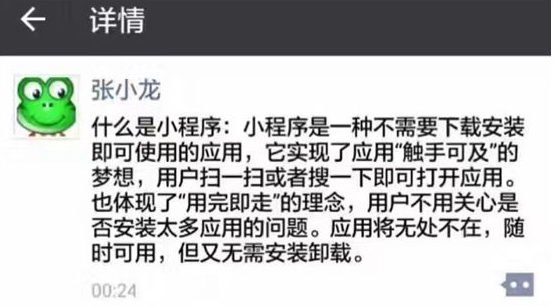
* 能够注册成功小程序账号
* 能够成功安装小程序开发工具(官方)
* 能够运行第一个简单的hello world小程序
* 掌握小程序文件结构
* 了解app.json配置文件与作用
* 了解小程序的生命周期
* 掌握wxml基本语法
* 掌握wxss基本样式语法

# 一、小程序介绍

## 1.1、小程序是什么

官方文档：<https://developers.weixin.qq.com/miniprogram/dev/>

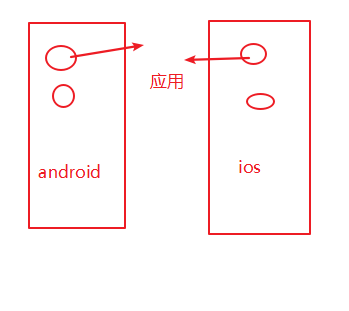
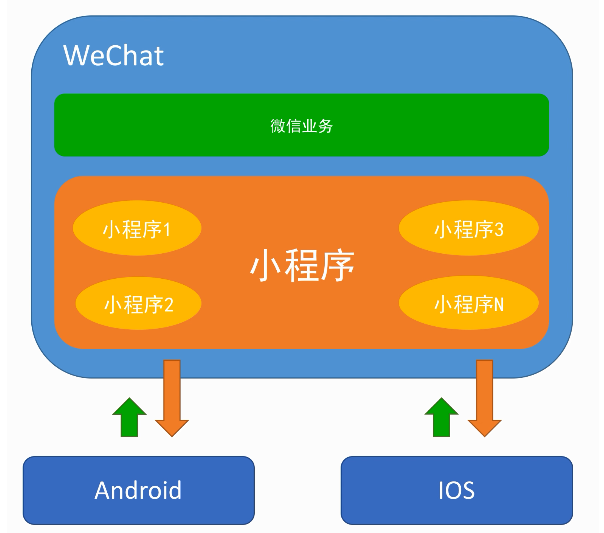
微信小程序，简称小程序，英文名 Mini Program，是一种不需要下载安装即可使用的应用，它实现了应用“触手可及”的梦想，用户扫一扫或者搜一下即可打开应用。也体现了“用完即走”的理念，用户不用关心是否安装太多应用的问题。应用将无处不在，随时可用，但又无需安装卸载。对于开发者而言，小程序开发门槛相对较低，难度不及APP，能够满足简单的基础应用。



## 1.2、如何理解小程序

* 不是HTML5
* 即用即走，随手可得
* 拥有离线能力
* 一次开发，多端兼容
* 依赖微信客户端中

## 1.3、小程序与传统App区别



小程序

局限性：必须依赖于微信，不能独立运行。

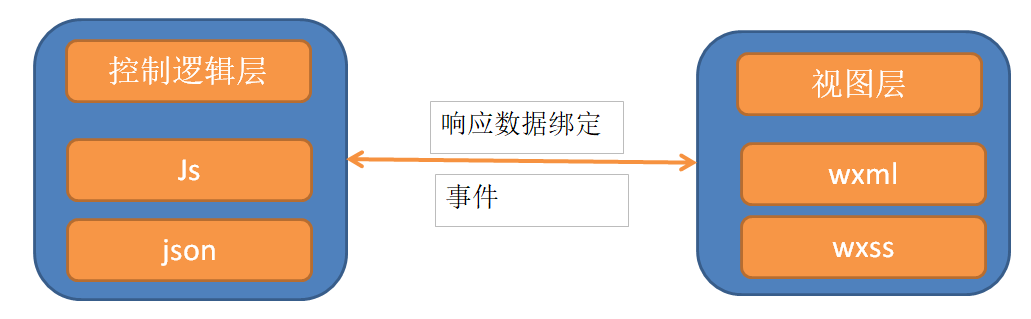
优势：不需要考虚兼容问题，安装的问题，开发难度小。

传统app：

优势：独立运行，不需要依赖于谁，可以适合所有的业务需求

局限性：需要用户安装，解决适配(开发兼容问题),开发难度大

## 1.4、小程序框架结构



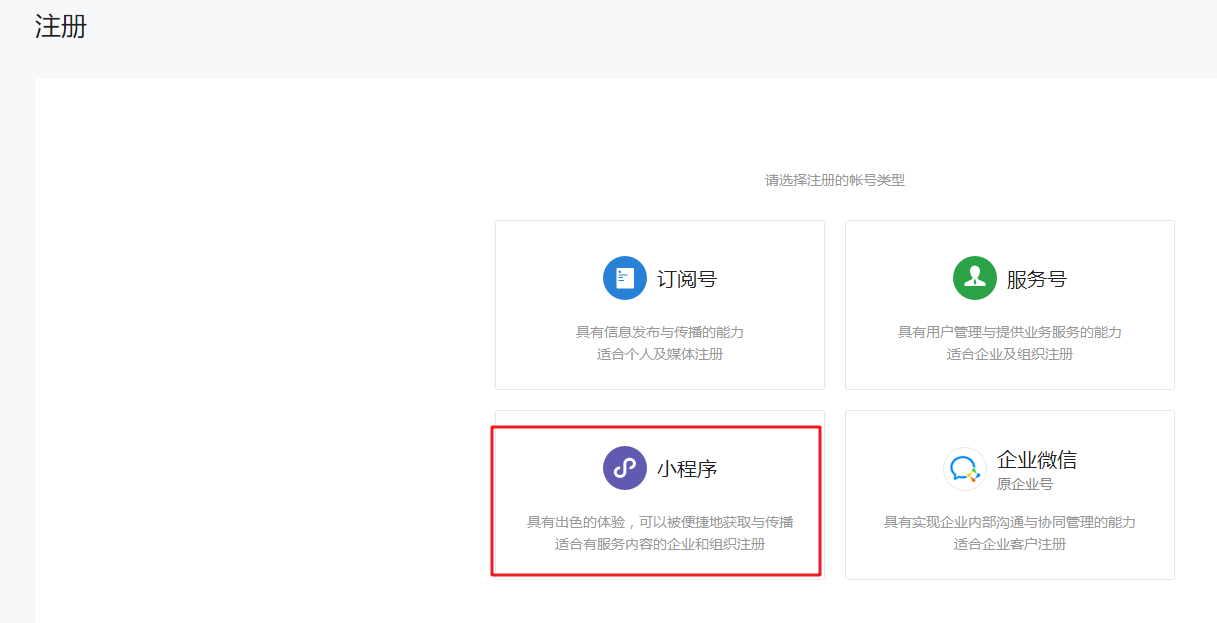
# 二、申请账号与登录设置

## 2.1、申请账号

官网: <https://mp.weixin.qq.com/>



选择小程序进行安装

## 2.3、登录

## 2.4、获取开发appid



有此appid就可以进行小程序的开发工作。

# 三、开发工具

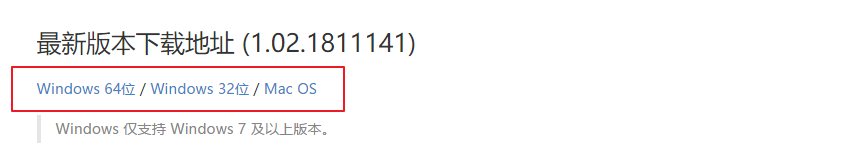
## 3.1、概览

为了帮助开发者简单和高效地开发和调试微信小程序，推出了小程序开发者工具，集成了公众号网页调试和小程序开发与调试两种开发模式。

## 3.2、下载安装

* 官方开发调式工具

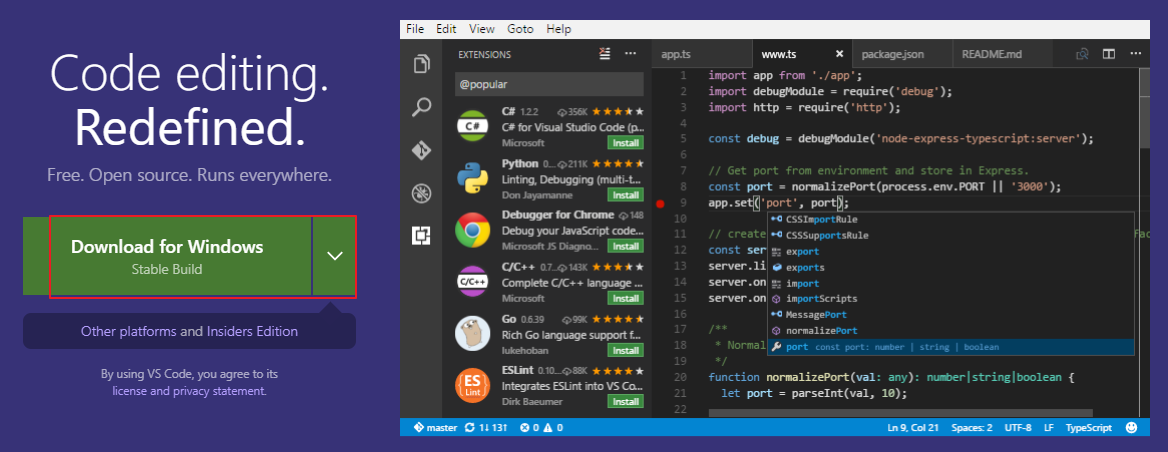
下载地址：<http://t.cn/RrKI5a3>



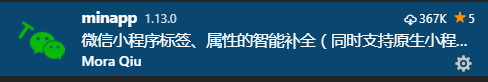
在windows下面安装的方式和其他的普通软件安装方式一致，一路下一步即可。

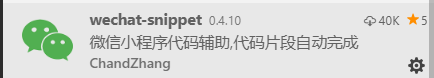
* vscode开发工具

下载地址：<https://code.visualstudio.com/>



vscode安装成功后，开发小程序需要安装2个插件





## 3.3、第一个小程序helloworld

创建小程序的应用，必须要用到appid。



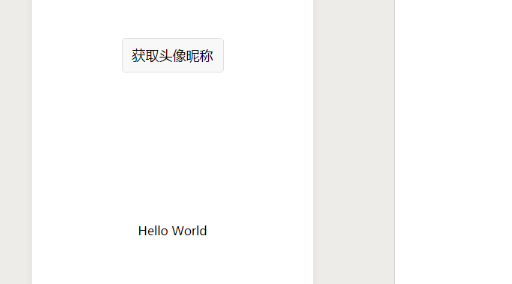
测试appid创建



正式appid创建



效果



# 四、小程序的文件结构及配置

## 4.1、文件结构

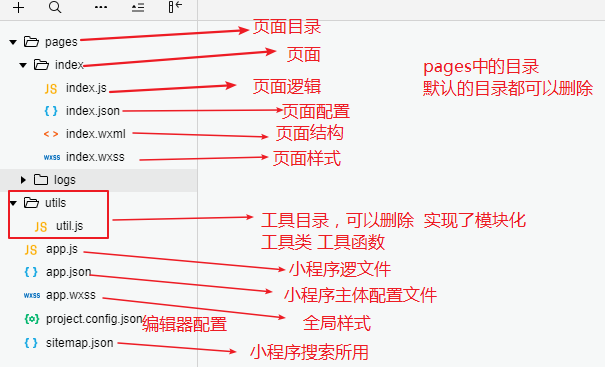
小程序包含一个描述整体程序的 app 和多个描述各自页面的 pages。

一个小程序主体部分由三个文件组成，必须放在项目的根目录，如下：

|  |  |  |
| --- | --- | --- |
| 文件 | 必填 | 作用 |
| app.js | 是 | 小程序逻辑 |
| app.json | 是 | 小程序公共设置 |
| app.wxss | 否 | 小程序公共样式表 |

一个小程序page页面由四个文件组成，分别是：

|  |  |  |
| --- | --- | --- |
| 文件类型 | 必填 | 作用 |
| js | 是 | 页面逻辑 |
| wxml | 是 | 页面结构 |
| wxss | 否 | 页面样式表 |
| json | 否 | 页面配置 |



## 4.2、小程序配置 app.json

参考网址：<https://developers.weixin.qq.com/miniprogram/dev/framework/config.html>

app.json 文件用来对微信小程序进行全局配置，决定小程序，页面数量、窗口表现、设置网络超时时间、设置底部或顶部菜单等。

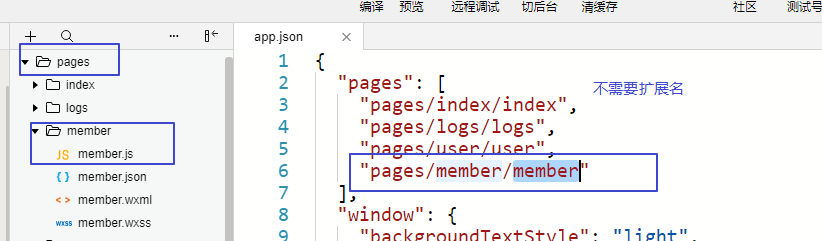
app.json中不能添加任何注释，key和value字符串必须用**双**引号引起来，数组或对象最后一位不能有逗号。

### 4.2.1、app.json 配置项列表

|  |  |  |  |
| --- | --- | --- | --- |
| 属性 | 类型 | 必填 | 描述 |
| pages | String Array | 是 | 设置页面路径 |
| window | Object | 否 | 设置默认页面的窗口表现 |
| tabBar | Object | 否 | 设置底部或顶部菜单 |
| networkTimeout | Object | 否 | 设置网络超时时间 |

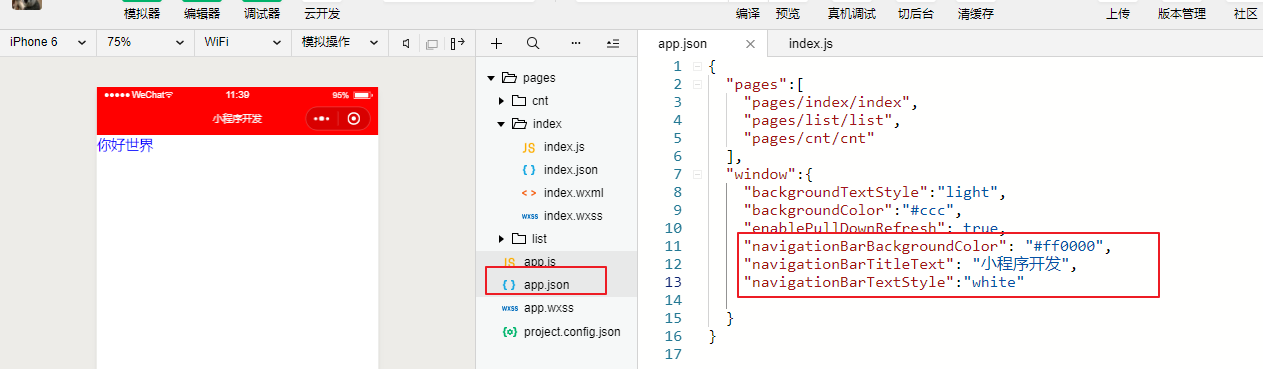
### 4.2.2、pages

指定小程序由哪些页面组成。每一项代表对应页面【路径+目录名+文件名(不包含后缀名)】，数组的第一项代表小程序的初始页面(首页/展示页)。小程序中新增/减少页面，都需要对 pages 数组进行修改。



### 4.2.3、window

用于设置小程序的状态栏、导航条、标题、窗口背景色。



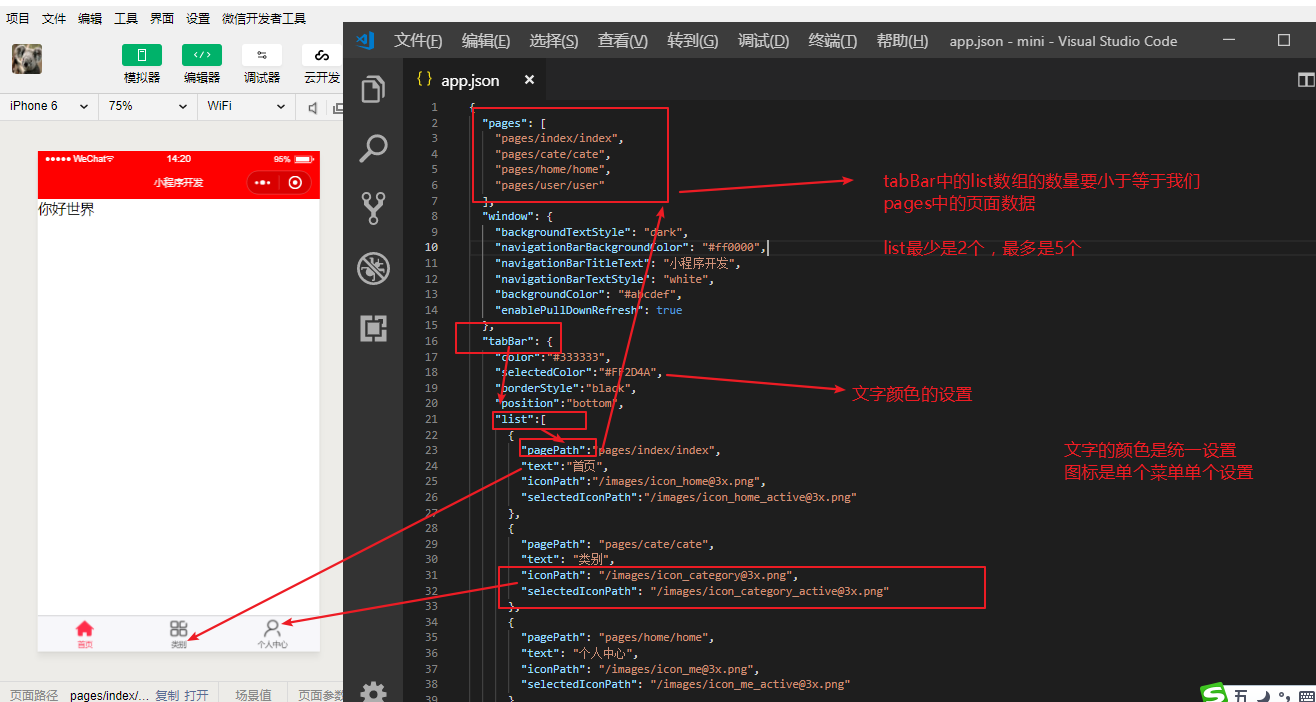
### 4.2.4、tabBar

设置小程序底部或顶部菜单栏。

注：

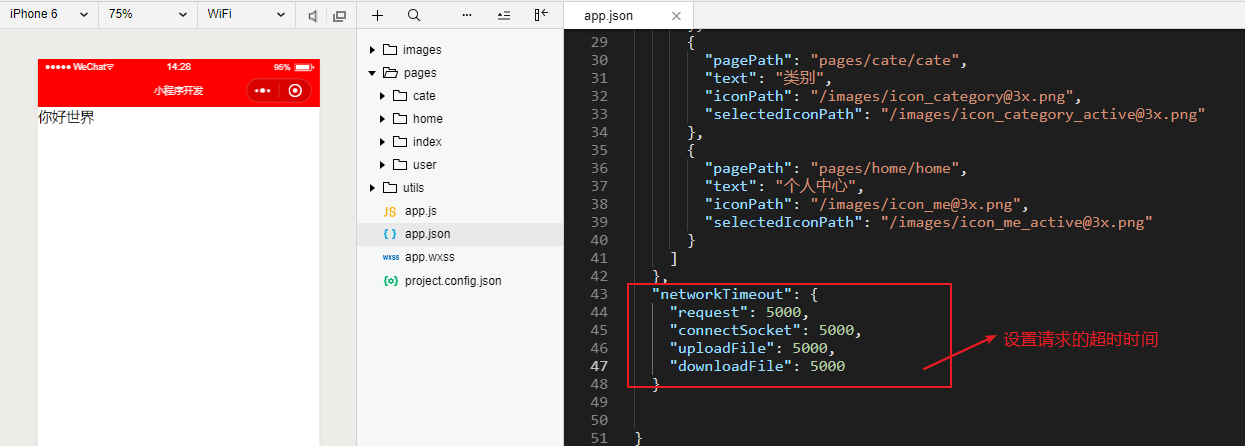
* 当设置 position 为 top 时，将不会显示 icon图标
* tabBar 中的 list 是一个数组，只能配置最少2个、最多5个菜单





### 4.2.5、networkTimeout

可以设置各种网络请求的超时时间。单位毫秒



# 五、生命周期

网址：<https://dwz.cn/OOHwXDNE>

由框架自己触发的一系统事件函数。不需要人来参与

## onLoad(Object[json] query)

页面加载时触发。一个页面只会调用一次，可以在 onLoad 的参数中获取打开当前页面路径中的参数。

参数说明

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| query | Object | 打开当前页面路径中的参数 json对象 |

## onShow()

页面显示/切入前台时触发。一个页面可以触发很多次。

## onReady()

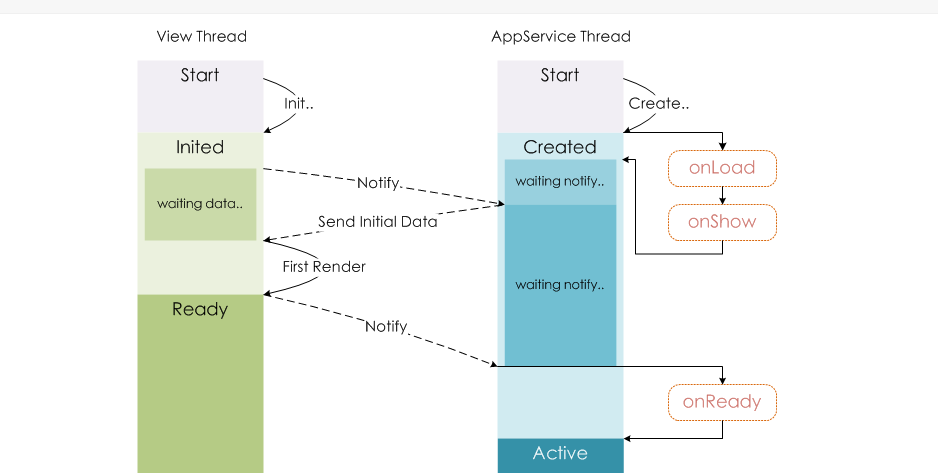
页面初次渲染完成时触发。一个页面只会调用一次。

## onHide()

页面隐藏/切入后台时触发。一个页面可以触发多次

## onUnload()

页面卸载时触发。



# 六、视图结构 wxml

## 6.1、wxml概述

从事过网页编程的人知道， HTML 是用来描述当前这个页面的结构，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。

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只需要负责数据的处理，然后再通过一种模板语法来进行界面结构展示。

在视图层中通过 **{{}}**语法把一个变量绑定到视图界面上，称为数据绑定

<view>{{变量}}</view>

当然仅仅通过数据绑定还不够的，还需要 if/else, for等控制能力，在小程序里边，这些控制能力都用 wx: 开头的属性来表达。wx:if wx:for

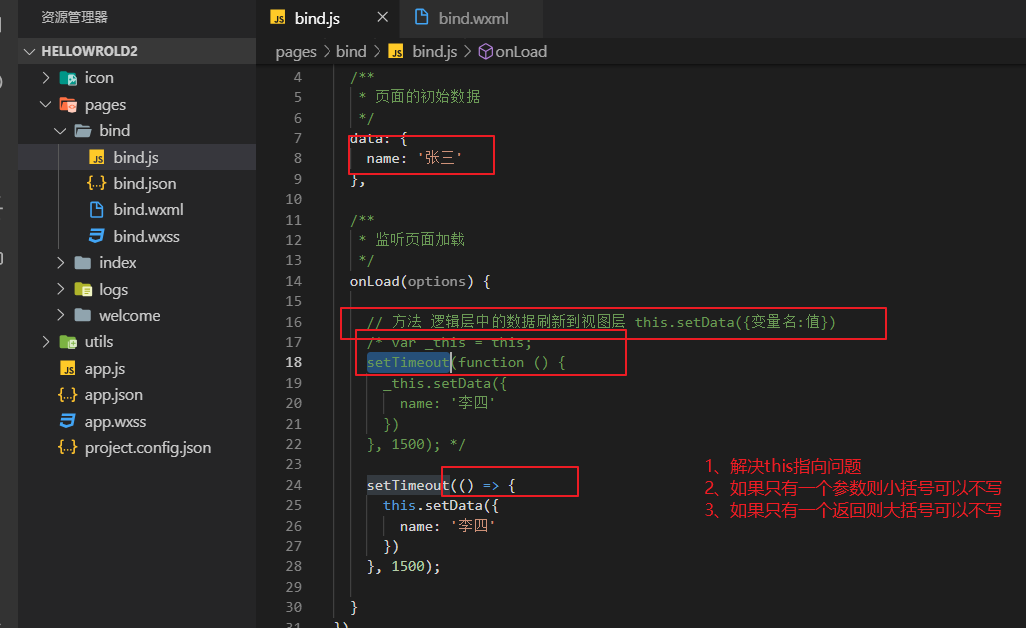
数据绑定

逻辑层刷新数据到视图层

this.setData({key:value})

一定要手动触发

js逻辑



wxml



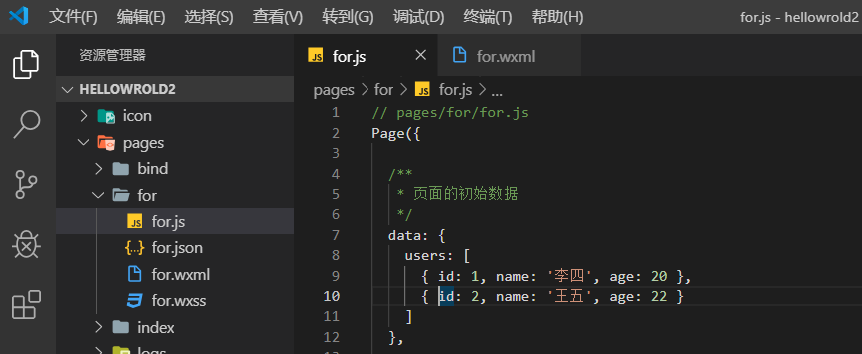


## 6.2、列表渲染 for

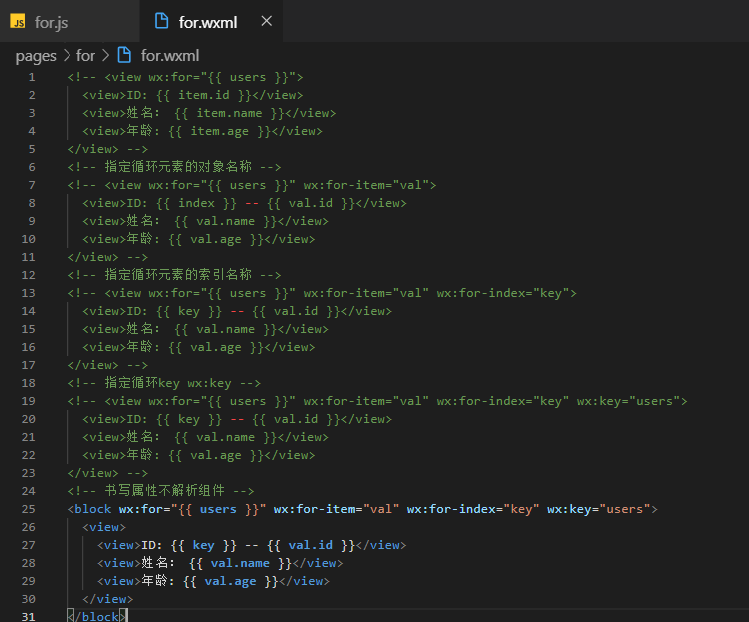
语法：wx:for 循环

* 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 默认名为 item
* wx:key 可以定义也可以不定义 唯一的标识符 提高性能

js逻辑



wxml循环





## 6.3、条件渲染

语法：wx:if

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：



## 6.4、引用(包含)

把模板定义到外部，然后多个页面间可以共用使用定义的模板WXML结构显示。

<https://developers.weixin.qq.com/miniprogram/dev/framework/view/wxml/import.html>

WXML 提供两种文件引用方式import和include





### 6.4.1、模板

定义用 template

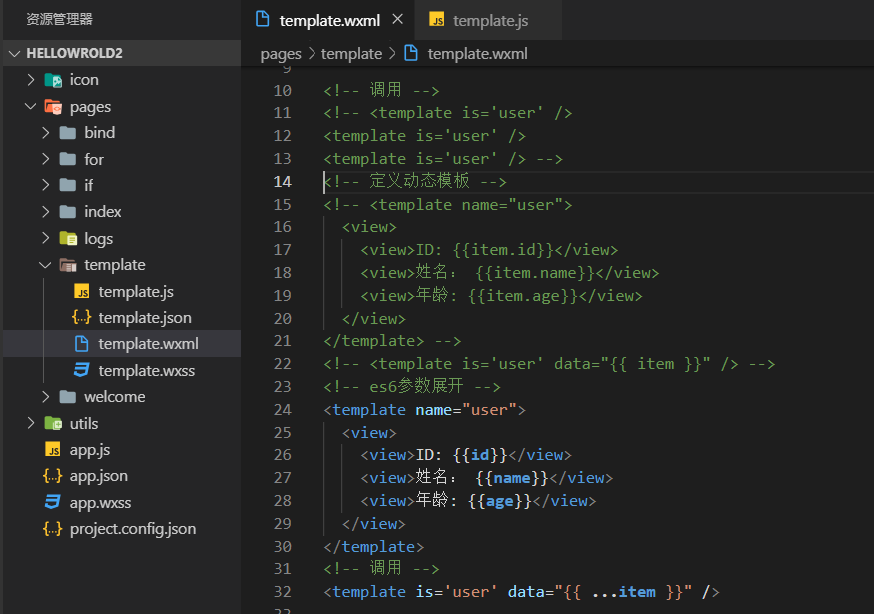
<template name=”名称”>

wxml语法

</template>

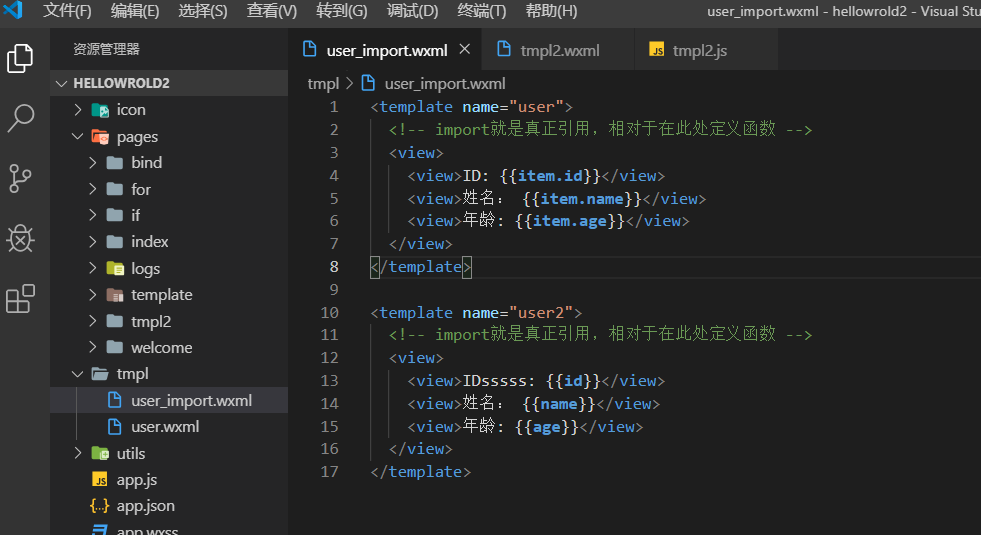
调用

<template is=”模板定义时name名称” [data=”{{传给模板的数据参数}}”] />

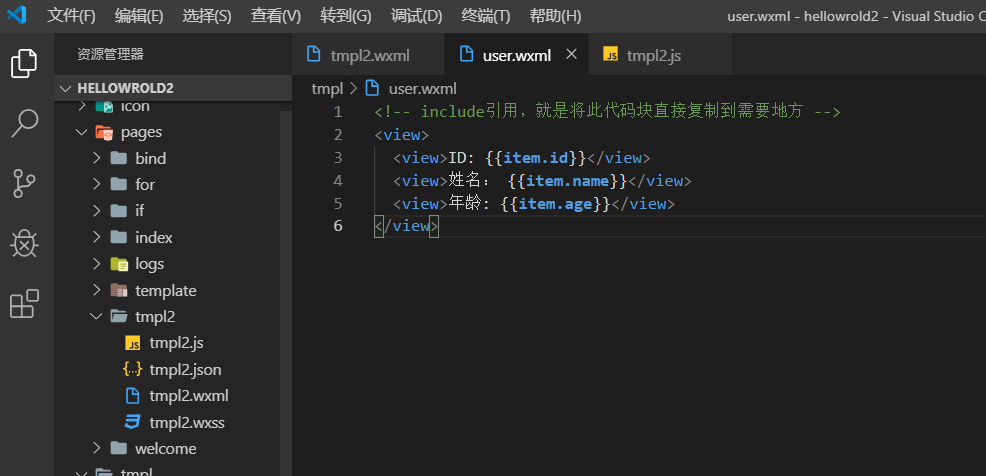


* import和include区别
* 定义方式不一样
* 引用的方式不一样，include 相当于复制代码过来，*import*相当于调用函数
* 数据的传递不一样 include 直接获取js中的数据，import必须通过template date属性获取

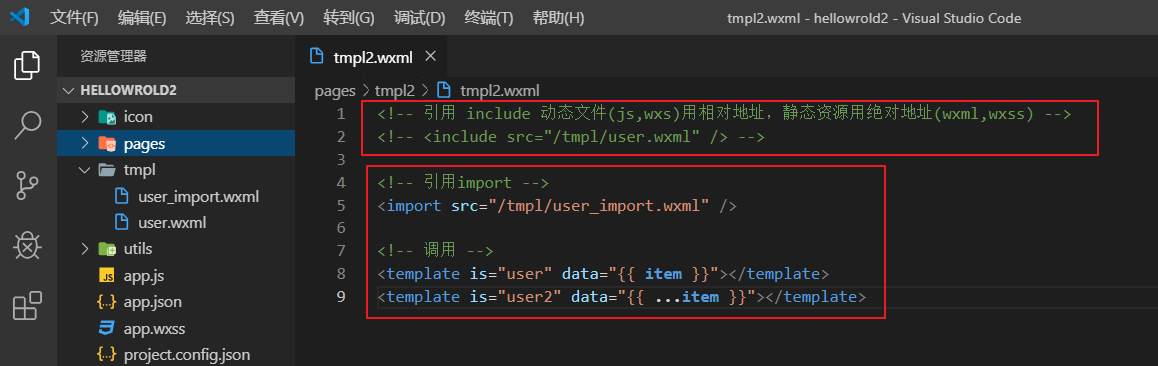
import定义的外部模板



include定义的名部模板



wxml中调用



# 七、小程序的样式

## 7.1、概述

WXSS 用来决定 WXML 的组件应该怎么显示。说白了就是样式

为了适应广大的前端开发者，WXSS 具有 CSS 大部分特性。同时为了更适合开发微信小程序，WXSS 对 CSS 进行了扩充以及修改。

* 新增了尺寸单位

WXSS 在底层支持新的尺寸单位 rpx 响应式尺寸单位

小程序中全屏尺寸数值是 : 750rpx

* 提供了全局的样式和局部样式

app.wxss 作为全局样式

局部页面样式 page.wxss 仅对当前页面生效 app.wxss<page.wxss<行级

* 此外 WXSS 仅支持部分 CSS 选择器

## 7.2、选择器

目前支持的选择器有：

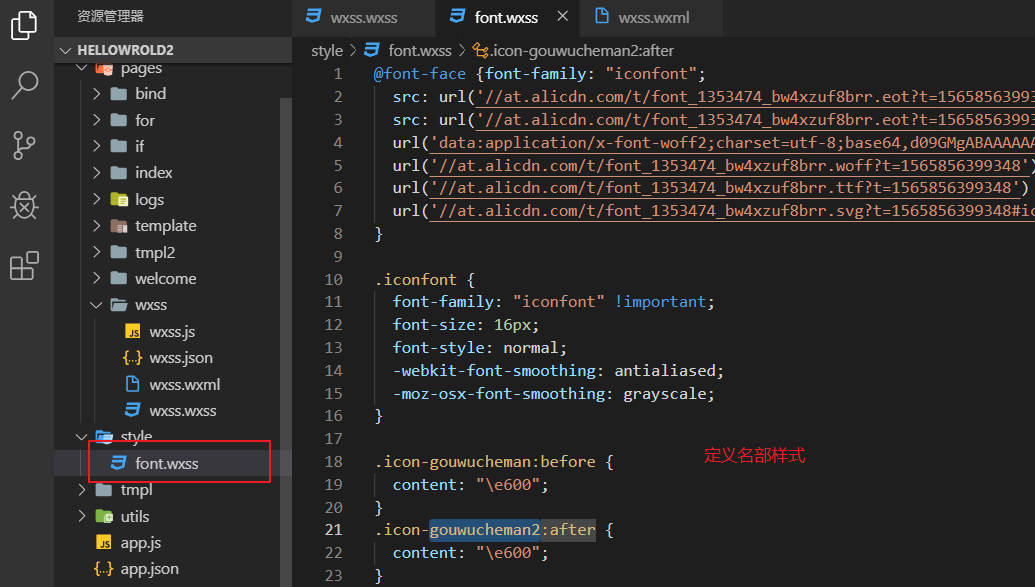
|  |  |  |
| --- | --- | --- |
| 选择器 | 样例 | 样例描述 |
| .class | .intro | 选择所有拥有 class="intro" 的组件 |
| #id | #firstname | 选择拥有 id="firstname" 的组件 |
| element | view | 选择所有 view 组件 |
| element, element | view, checkbox | 选择所有文档的 view 组件和所有的 checkbox 组件 |
| ::after | view::after | 在 view 组件后边插入内容 |
| ::before | view::before | 在 view 组件前边插入内容 |

## 7.3、样式导入

使用 @import 语句可以导入外联样式表，@import 后跟需要导入的外联样式表的相对路径，用 ; 表示语句结束。

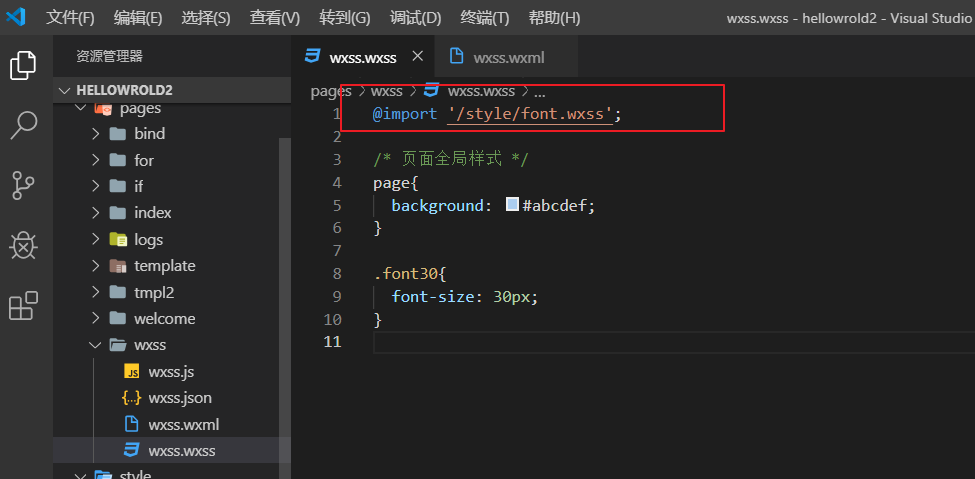
在样式中引入@import 路径;

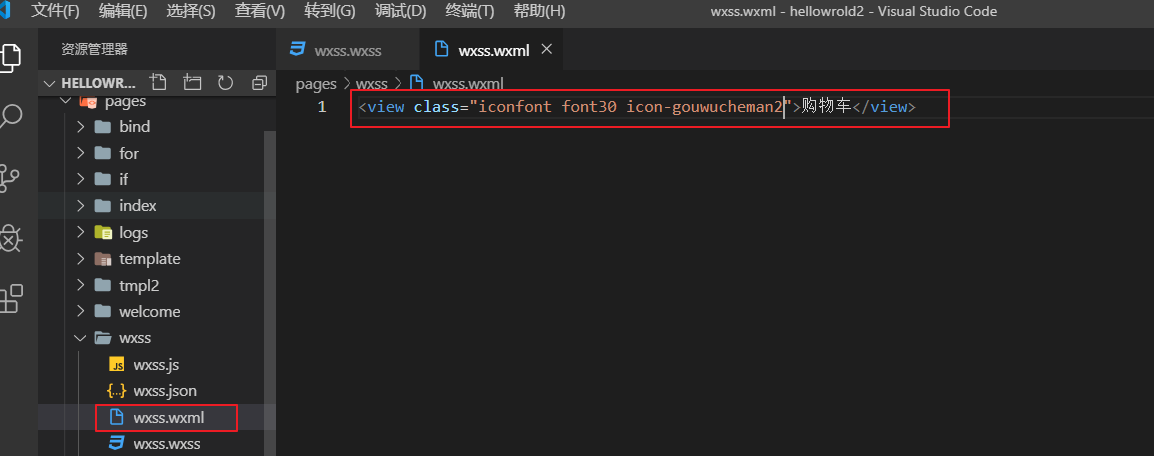
第1步：定义外部样式



第2步：在需要引用样式的文件中引入

@import 绝对地址





# 八、flex

## 8.1、概述

**弹性布局**

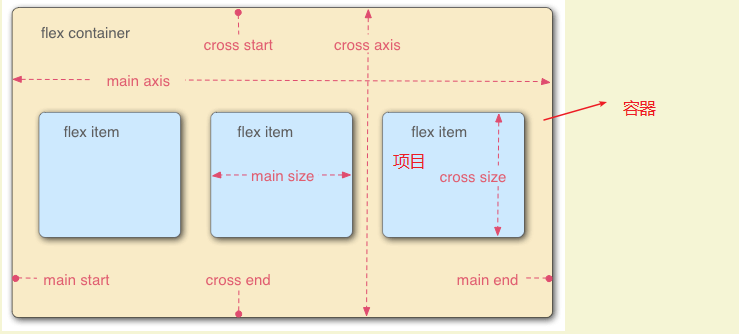
w3c在2009年提出来一个新的布局标准，也是现在目前主流的布局标准。

Flex 布局，可以简便、完整、响应式地实现各种页面布局。目前，它已经得到了所有浏览器的支持，这意味着，现在就能很安全地使用这项功能。



## 8.2、容器

主轴和交叉轴



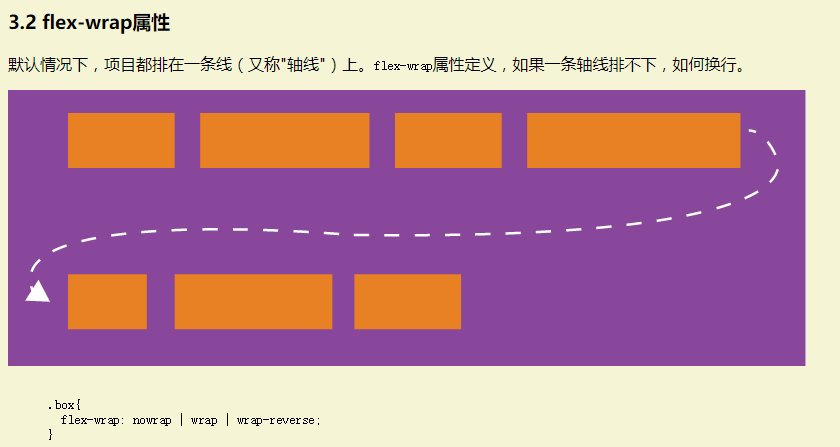
默认是以水平(x轴)为主轴，垂直(Y轴)为交叉轴

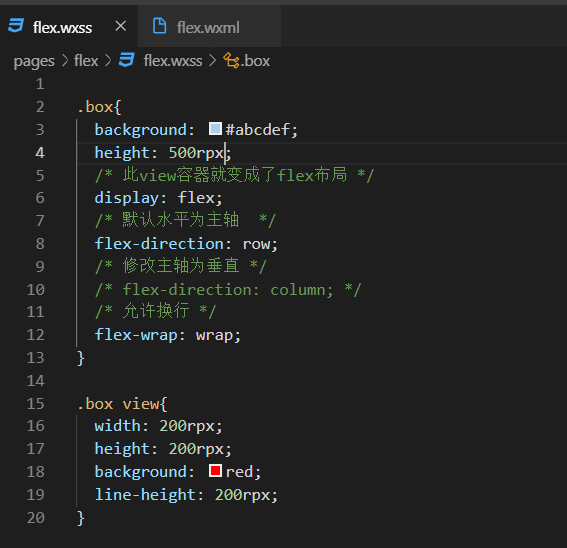
注意，设为 Flex 布局以后，子元素的float、clear和vertical-align属性将失效。

定位是不受影响 flex+position

### 8.2.1、flex-direction属性







### 8.2.2、主轴和交叉轴对齐方式

###### 主轴

justify-content: flex-start | flex-end | center | space-between | space-around;

flex-start（默认值）：左对齐

flex-end：右对齐

center： 居中

space-between：两端对齐，项目之间的间隔都相等。

space-around：每个项目两侧的间隔相等。所以，项目之间的间隔比项目与边框的间隔大一倍。

###### 交叉轴

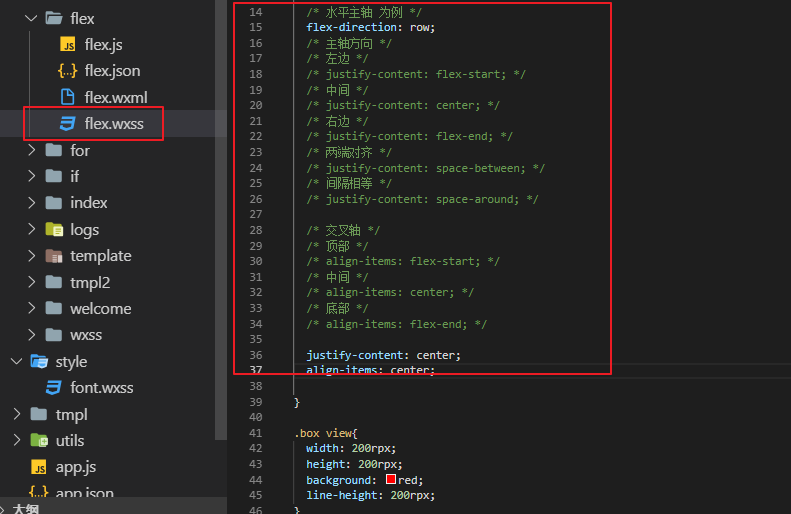
align-items: flex-start | flex-end | center

flex-start：交叉轴的起点对齐。

flex-end：交叉轴的终点对齐。

center：交叉轴的中点对齐。

wxss



wxml

