1,view 页面渲染，视图，类似前端，一个线程

2，App Service 数据请求，接口调用，逻辑处理 类似后端， 通知到视图层（更新等，通知逻辑层） 一个线程

1.view视图：WXML WXSS 由组件

WXML:数据绑定，逻辑运算，模板，添加事件

wxml编译器：wcc 把wxml文件 转为 js 执行方式：wcc index.wxml

2.view WXSS (支持绝大部分css)

wxss编译器：wcsc 把wxss文件转化为 js 执行方式： wcsc index.wxss

3.WXSS Selectors

4、View – Component

5、View – Native Component

（目前Native实现的组件有 <canvas/> <video/> <map/> <textarea/>）

6.App Service (逻辑层）

App()小程序入口，page()入口 数据绑定、事件分发、生命周期管理、路由管理

7.App Service – Binding

数据绑定使用 Mustache 语法（双大括号）将变量包起来，动态数据均来自对应 Page 的 data，可以通过setData方法修改数据。

事件绑定的写法同组件的属性，以 key、value 的形式，key 以bind或catch开头，然后跟上事件的类型，如bindtap, catchtouchstart，value 是一个字符串，需要在对应的 Page 中定义同名的函数

等功能

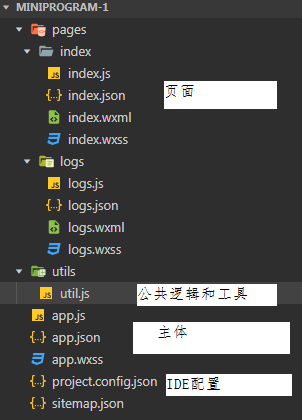
8.文件类型简介

.json为配置文件，例如：配置标题 "navigationBarTitleText": "登录"

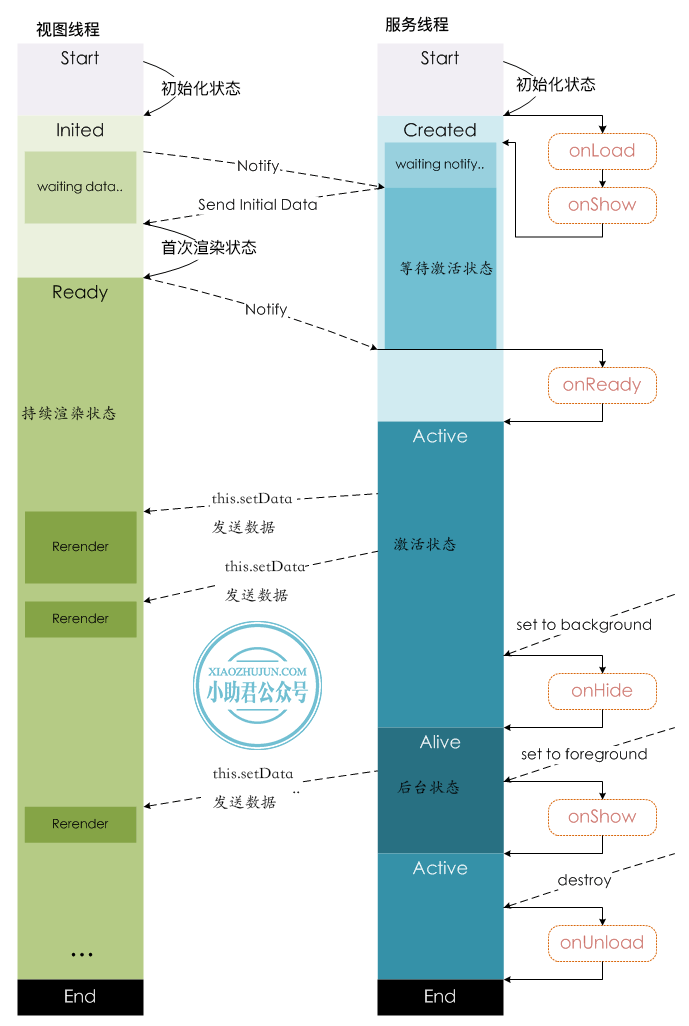
.wxml 为模板文件，相当于HTML模板，（做界面布局显示）

.wxss 为样式文件，相当于HTML的CSS样式表，（做界面样式的控制）

.js 为JS 脚本逻辑文件，相当于HTML的js脚本，（做功能的逻辑）



三.小程序“前端”VIEW与“后端”App Service 交互流程图



页面生命周期函数的调用顺序为：

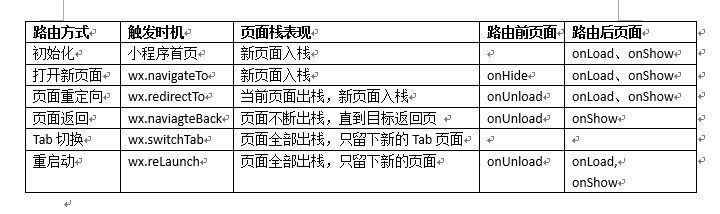
onLoad>onShow>onReady

基本上加载一个小程序进来之后的执行顺序为：

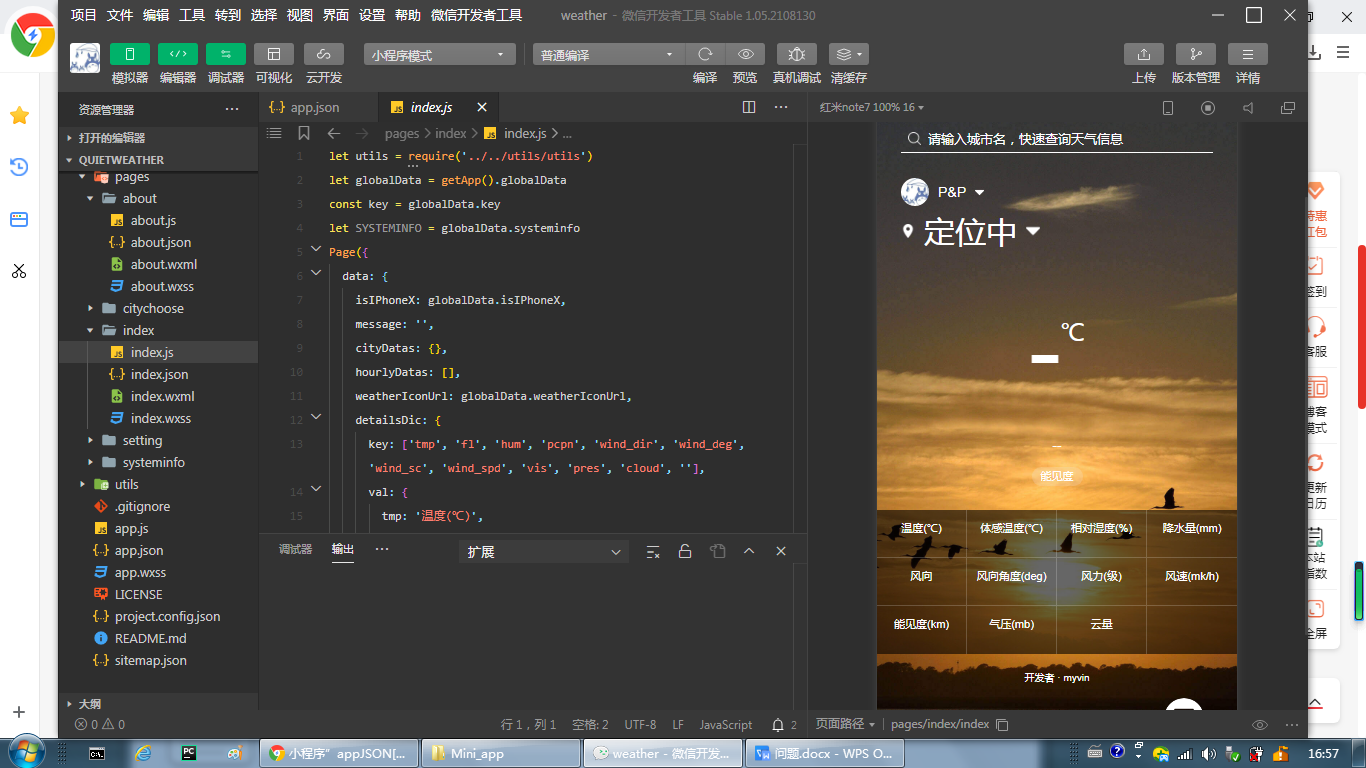
onLaunch–onLoad–onShow–onReady–onHide

小程序切换后台：

页面onHide-小程序隐藏onHide



5.找了个代码看了下效果：



1. 小程序的几种页面跳转方式：

****wx.navigateTo：****用于保留当前页面、跳转到应用内的某个页面（但是不能跳到tabbar页面），使用 wx.navigateBack可以返回到原页面

****wx.redirectTo：****用于关闭当前页面，跳转到应用内的某个页面。这样的跳转，可以避免跳转前页面占据运行内存（微信所限制的 5 层页面栈），但返回时页面需要重新加载

wx.reLaunch： wx.reLaunch()先关闭了内存中所有保留的页面，再跳转到目标页面。

****wx.switchTab：****它会先关闭所有非 tab bar 的页面。其次，也可以选择 wx.reLaunch()，它也能实现从非 tab bar 跳转到 tab bar，或在 tab bar 间跳转

****wx.navigateBack:****用于关闭当前页面，并返回上一页面或多级页面,若 delta 的取值大于现有可返回页面数时，则返回到用户进入小程序的第一个页面。当不填写 delta 的值时，就默认其为 1（注意，默认并非取 0），即返回上一页面。