



---

# 小程序开发入门教程

---

# 目录

## CONTENTS

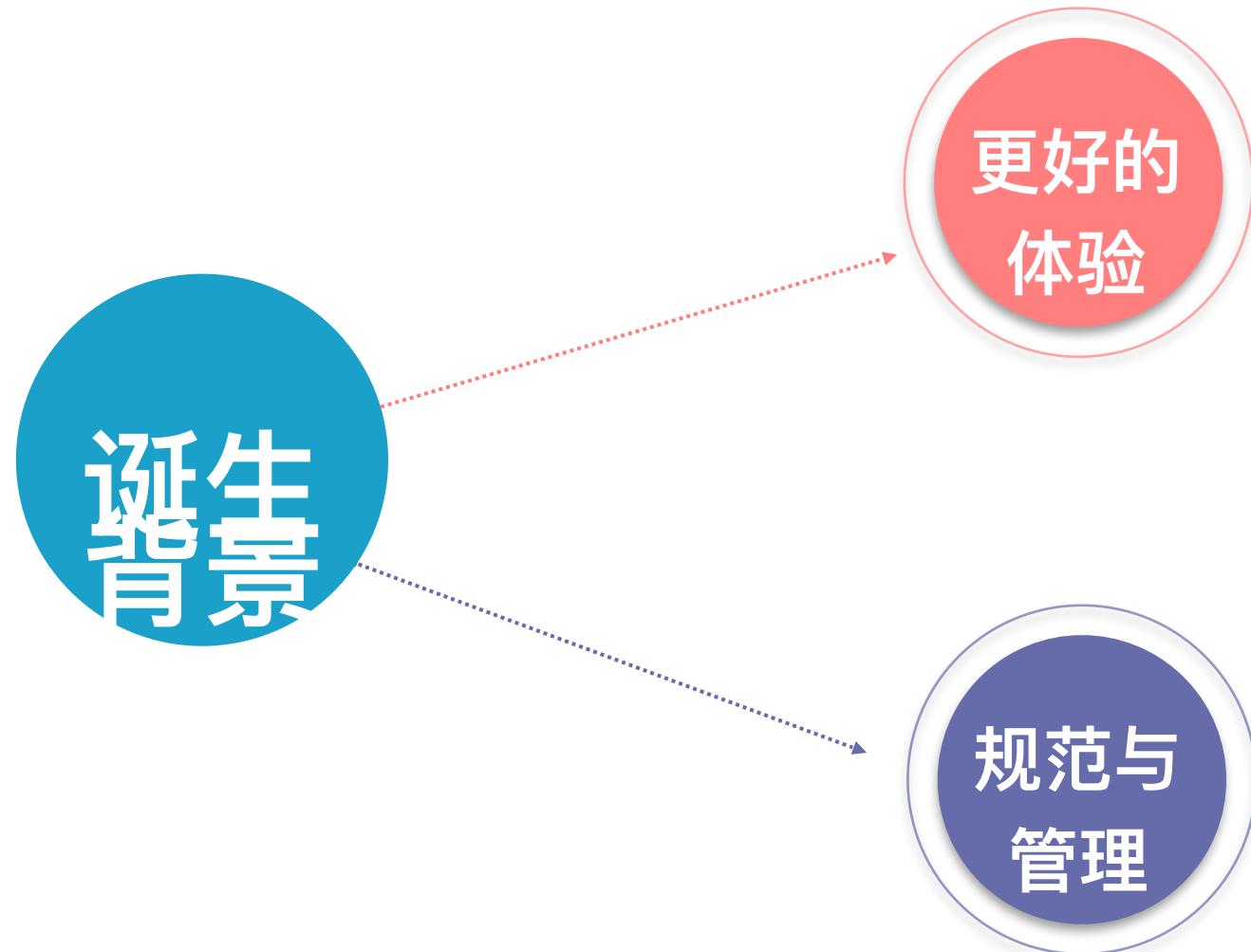
- 1 微信小程序介绍与准备
- 2 上手第一个小程序
- 3 微信小程序开发框架
- 4 微信小程序原生组件
- 5 微信小程序原生API
- 6 小程序开发核心技能
- 7 小程序开发项目实战案例
- 8 小程序开发实战注意事项与进阶指导

# 微信小程序介绍与开发准备

## 背景

- 1 为什么会有小程序?
- 2 什么是小程序?
- 3 小程序和移动应用程序区别
- 4 小程序能做什么?

# 微信小程序介绍与开发准备一为什么会有小程序



# 微信小程序介绍与开发准备—什么是小程序

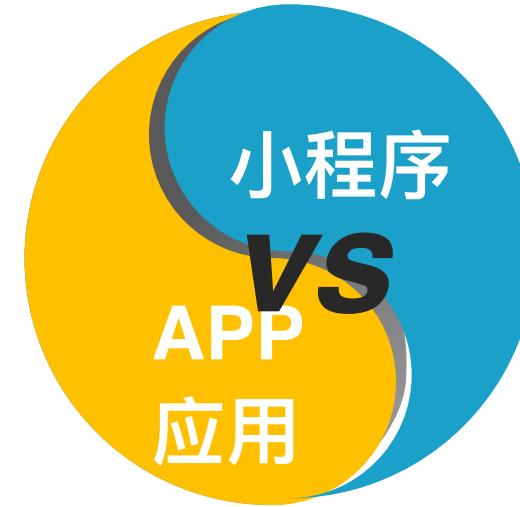
## 什么是小程序

小程序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

# 微信小程序介绍与开发准备—小程序与应用程序区别



无需安装

不占内存

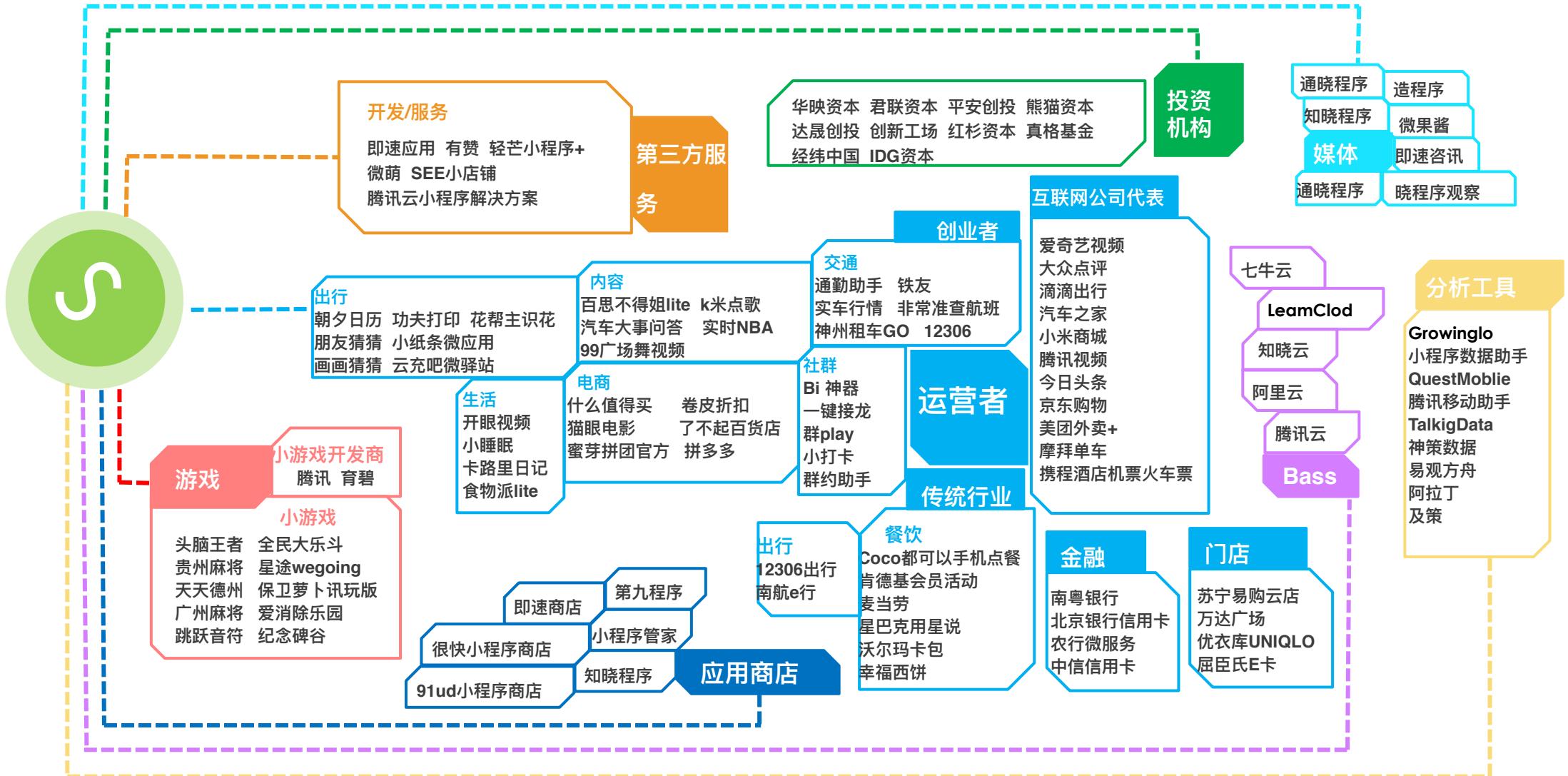
易传播

# 微信小程序介绍与开发准备—小程序能做什么



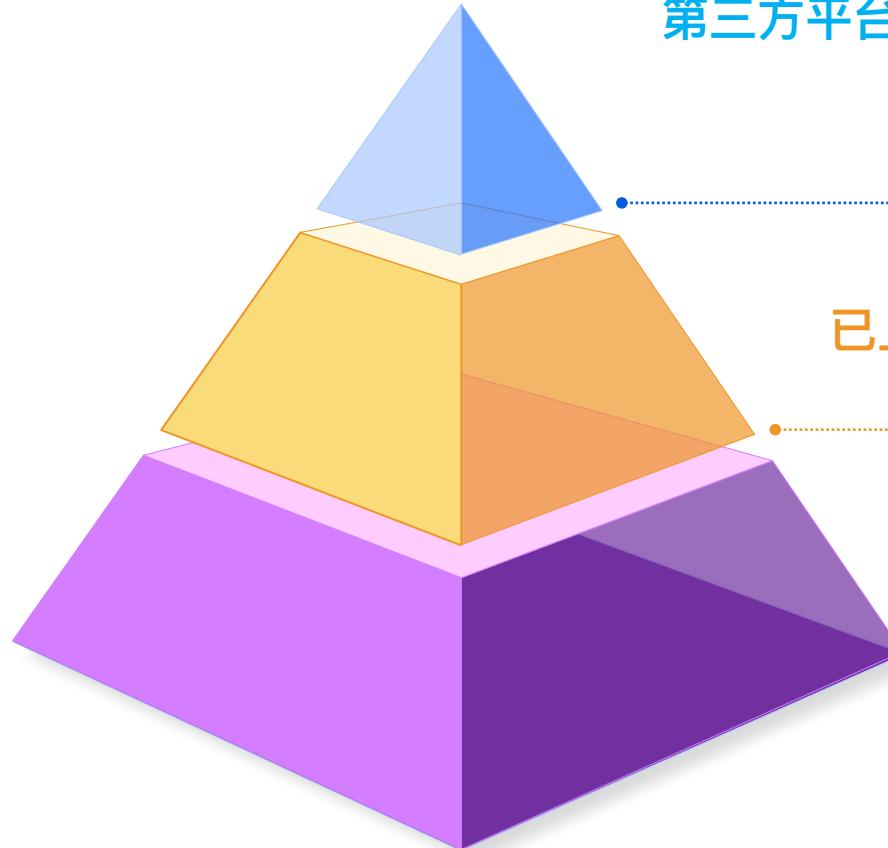
# 微信小程序介绍与开发准备—从业人员就业前景和发展

## 小程序生态云图



# 微信小程序介绍与开发准备—从业人员就业前景和发展

## 小程序生态



开发者总数：100万+

第三方平台：2000+

20岁以下占比：5.5%

海外开发者占比：1.1%

女性开发者：1.6%

已上线小程序数：58万+

日活跃用户数：1.7亿+

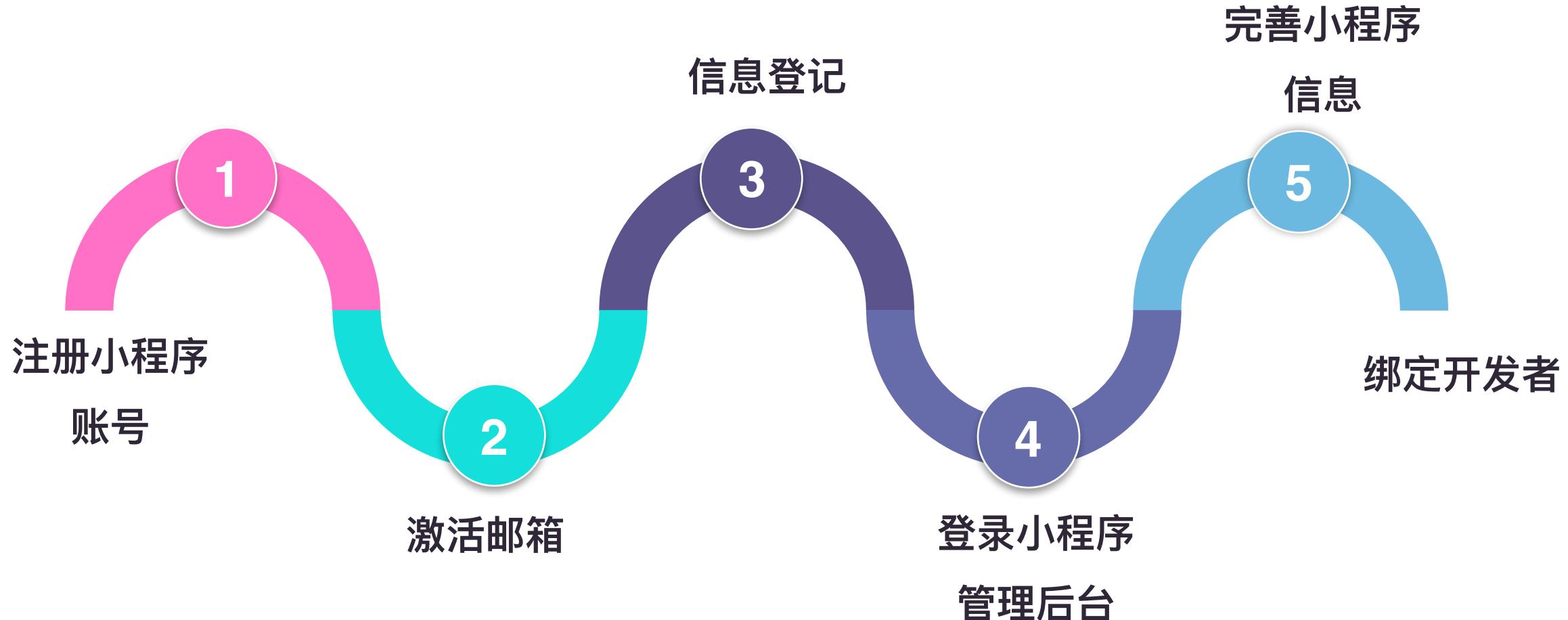
一线城市：30%

三线城市：19%

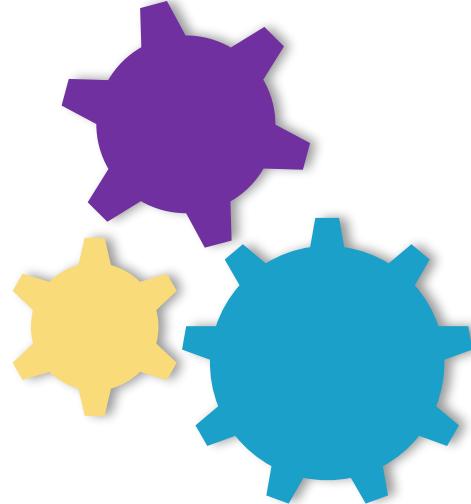
二线城市：20%

四线及以下：31%

# 微信小程序介绍与开发准备——小程序开发准备工作



# 微信小程序介绍与开发准备—安装使用微信开发者工具

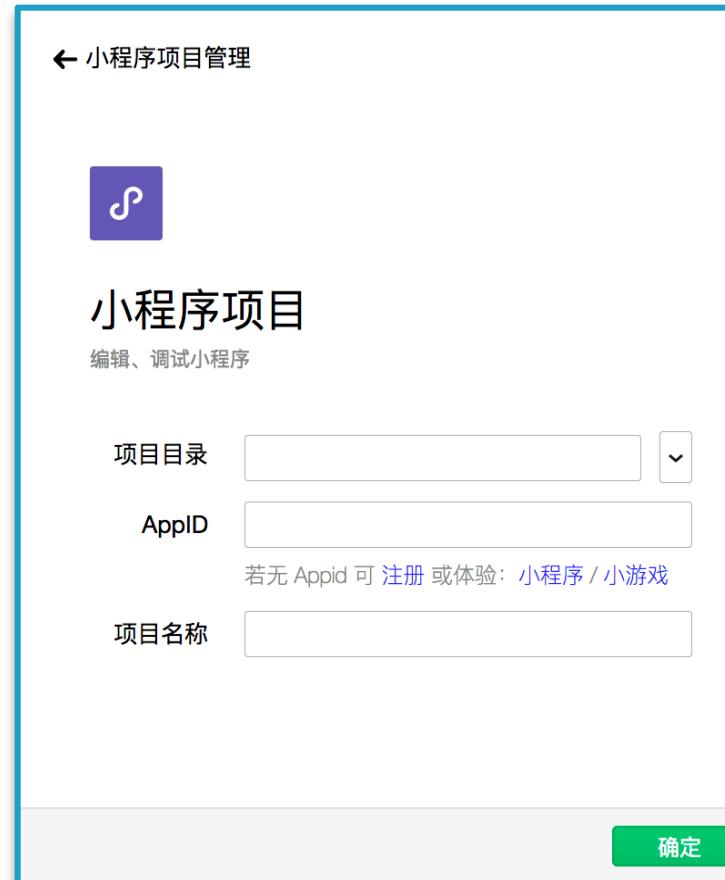


安装微信  
开发者工具

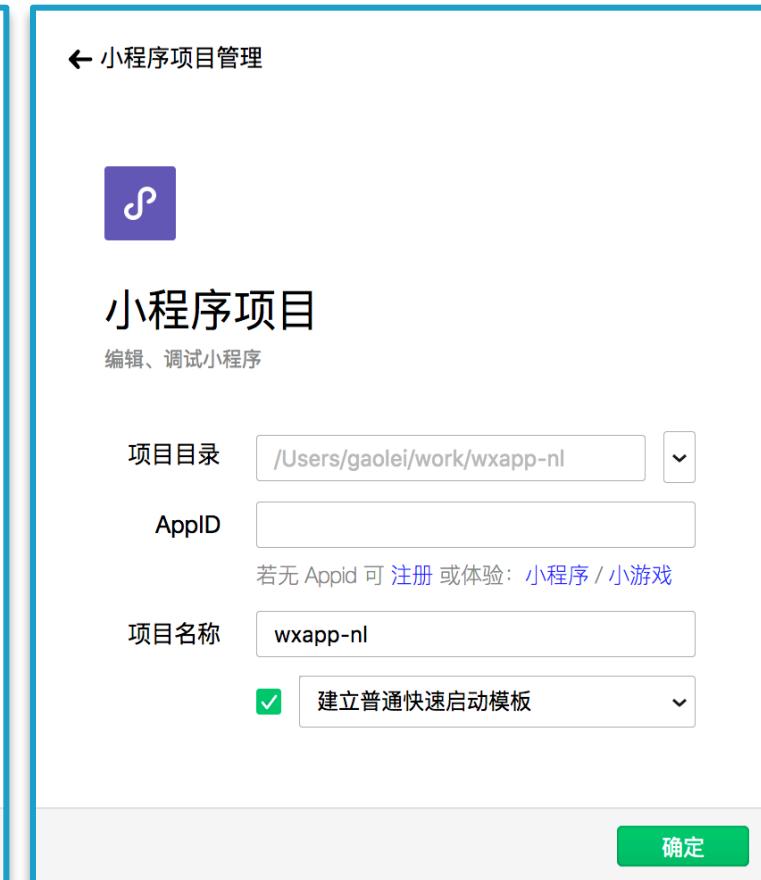
# 微信小程序介绍与开发准备—安装使用微信开发者工具



1



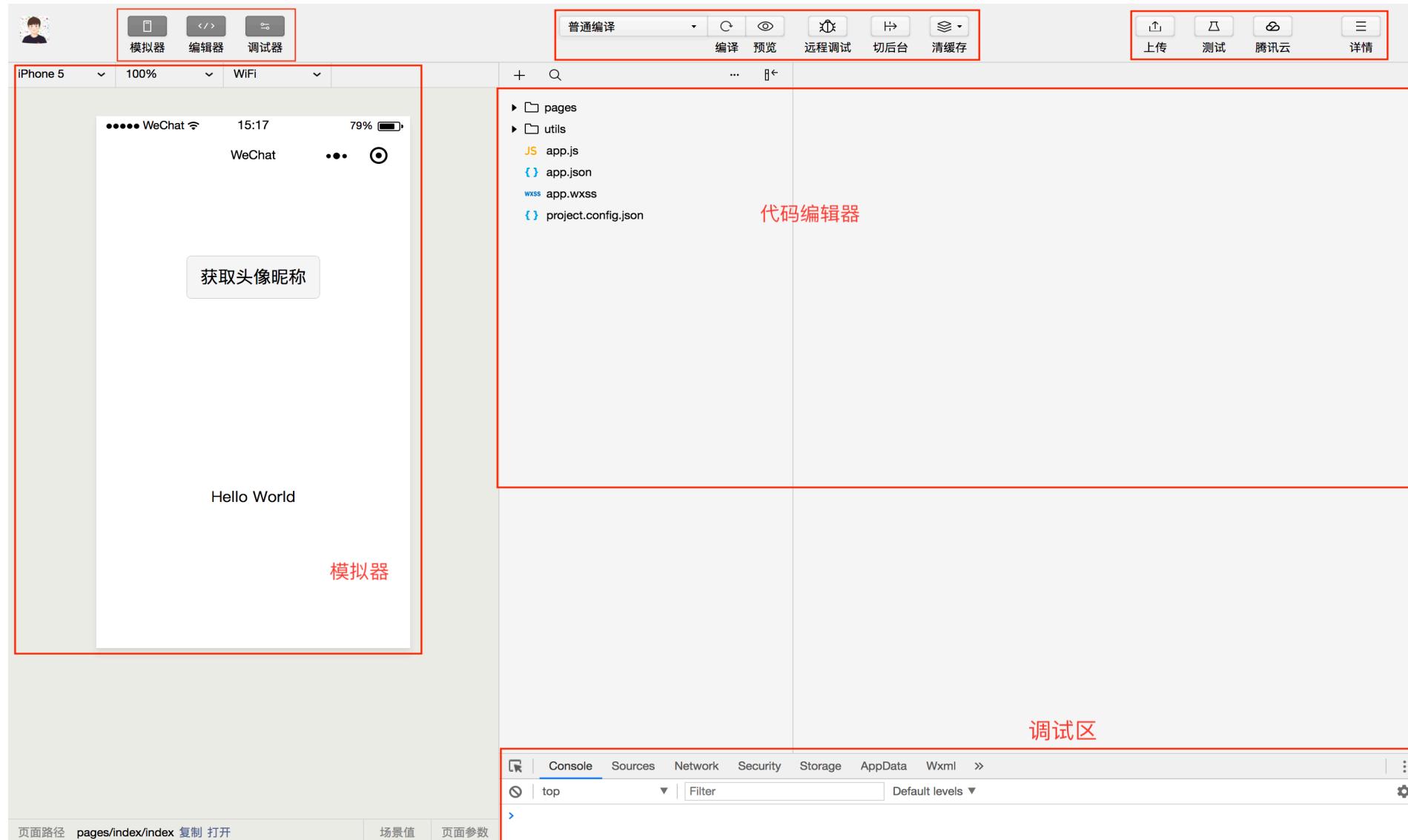
2



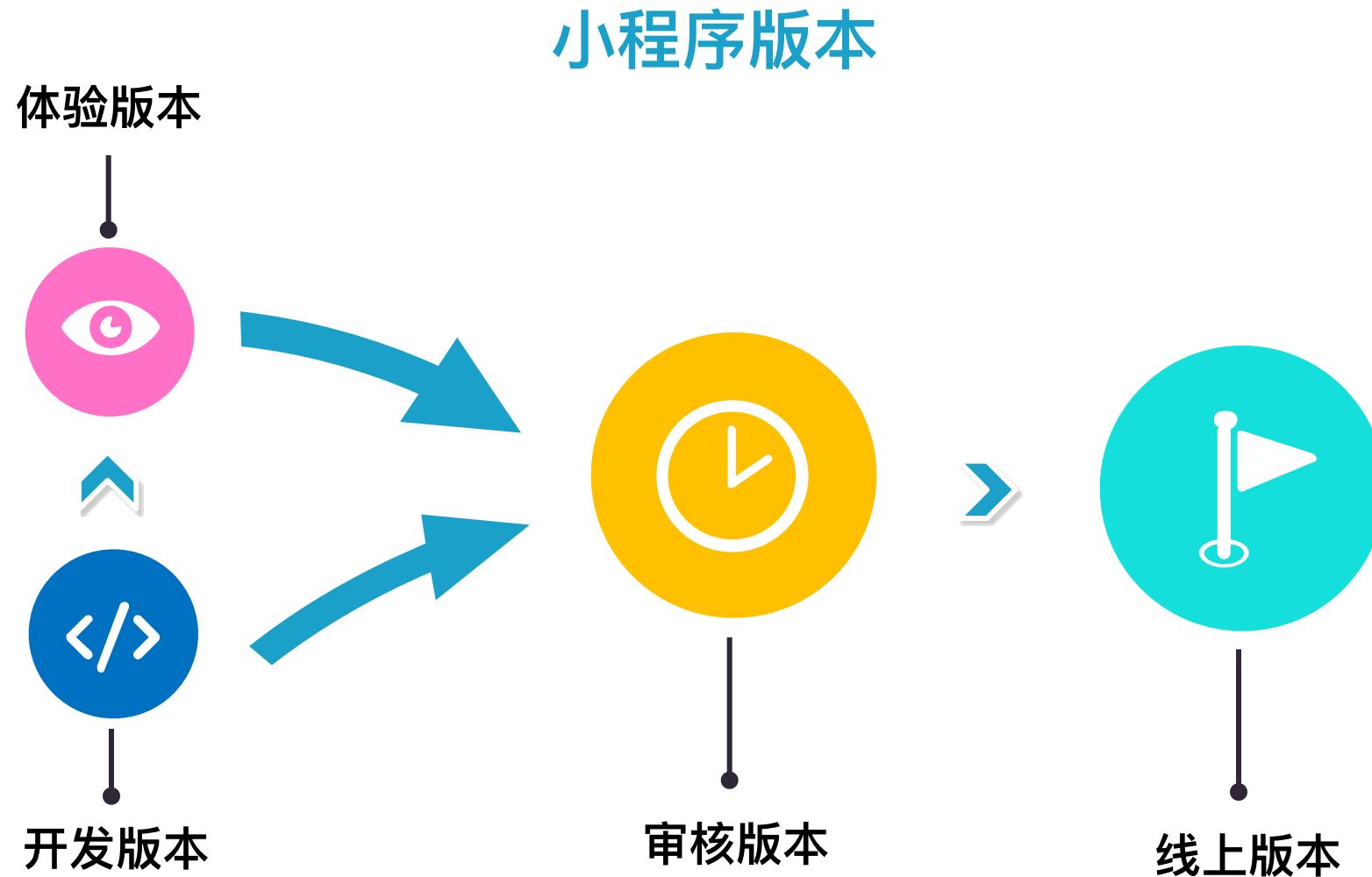
3



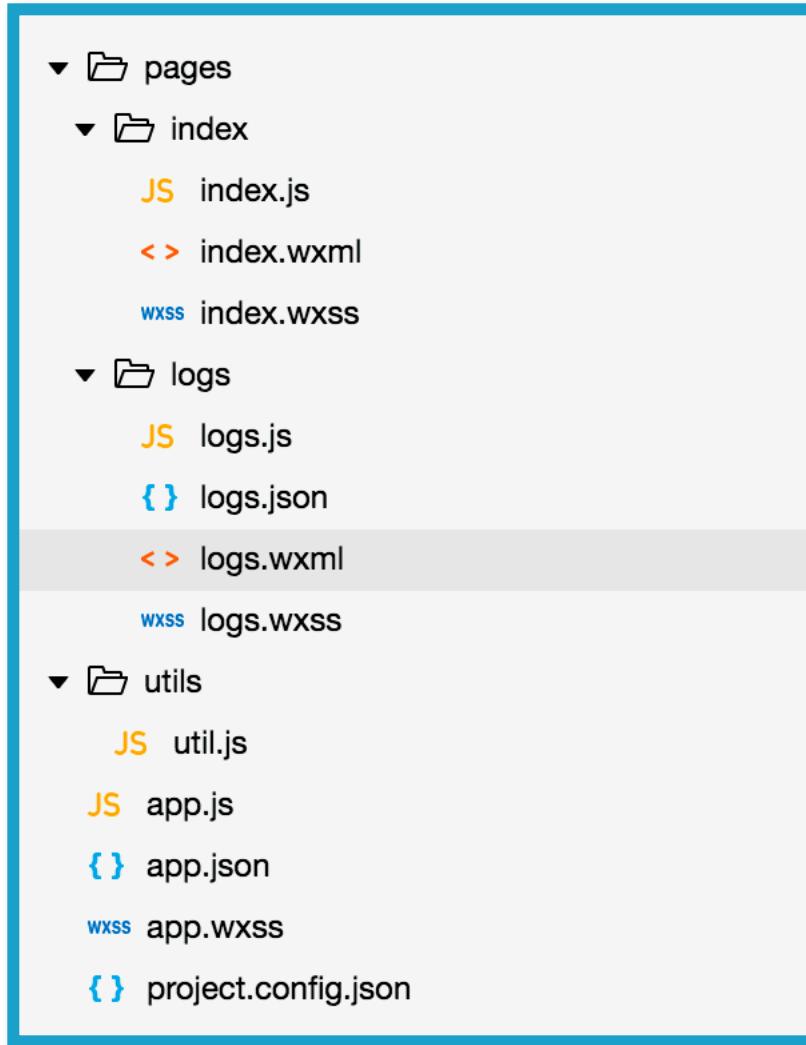
# 微信小程序介绍与开发准备—安装使用微信开发者工具



# 微信小程序介绍与开发准备—安装使用微信开发者工具



# 上手第一个小程序——小程序代码结构与基本配置



app.js



app.json



app.wxss



project.config.json



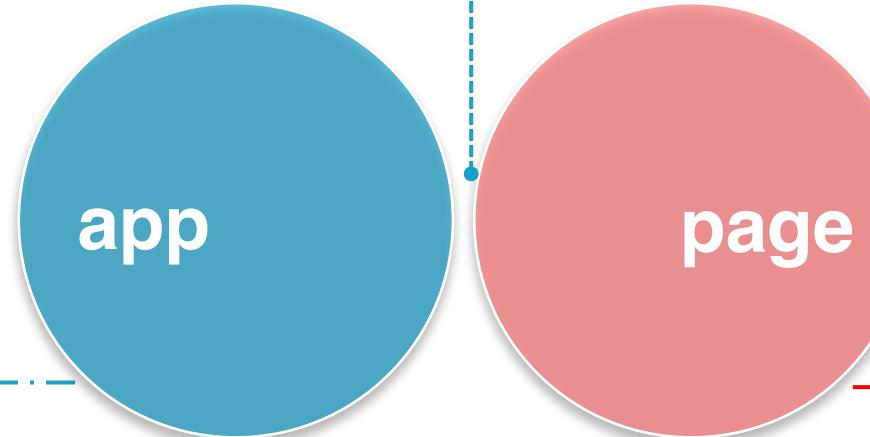
pages



utils

# 上手第一个小程序——小程序代码结构与基本配置

Pages  
tabBar  
networkTimeo  
debug  
navigationStyle



# 目录

## CONTENTS

3 微信小程序开发框架

4 微信小程序原生组件

5 微信小程序原生API

# 微信小程序开发框架——WXML



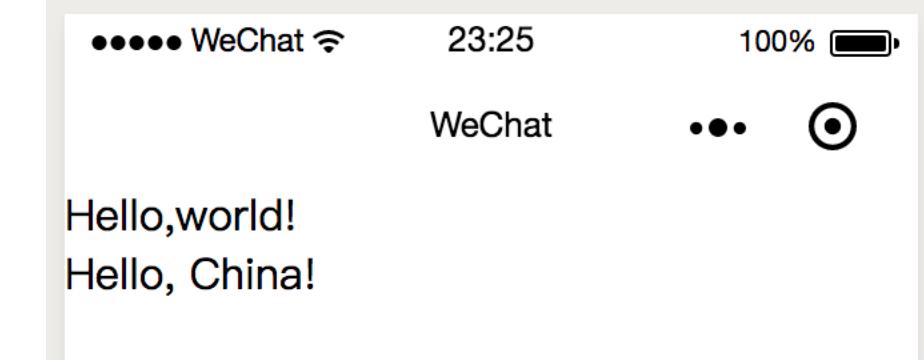
## WXML (WeiXin Markup Language)

是框架设计的一套标签语言，结合组件、  
WXS和事件系统，可以构建出页面的结构。

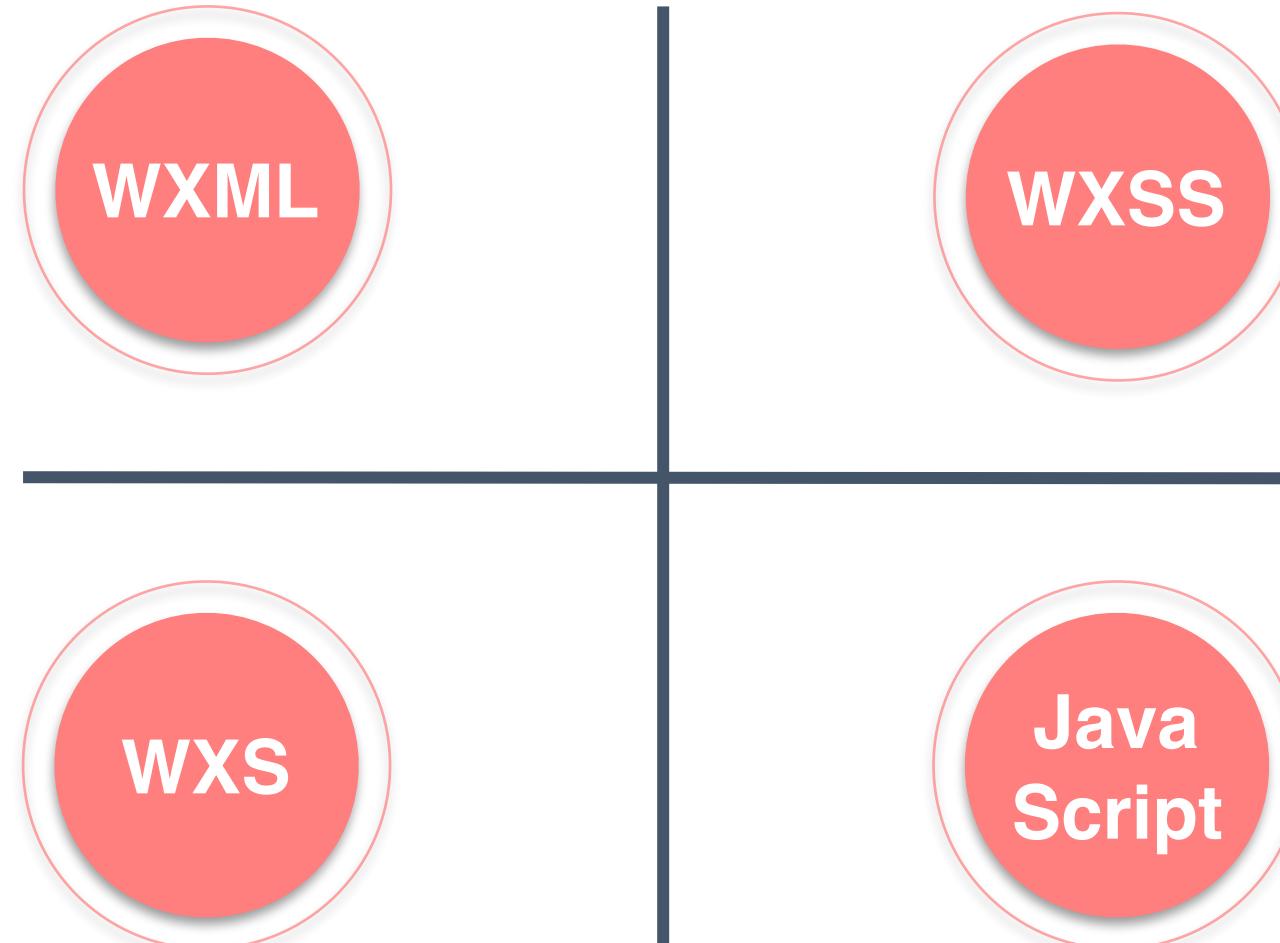
```
<标签名 属性名="属性名1" 属性名="属性名2" ...>
    ...
</标签名>
```

# 微信小程序开发框架——WXML

```
1  <!--index.wxml-->
2  <view class="className" data-name="A">
3      Hello,world!
4      <view>
5          Hello, China!
6      </view>
7  </view>
```



# 微信小程序开发框架——基本构成



# 微信小程序开发框架——WXML



-  数据绑定
-  列表渲染
-  条件渲染
-  模板引用

# 微信小程序开发框架——数据绑定

Mustache

```
1  <!--index.wxml-->
2  <view>
3  |   <text>{{message}}</text>
4  </view>
5  // index.js
6  Page({
7  |   data: {
8  |     message: "Hello, world"
9  |   }
10 })
```



# 微信小程序开发框架——数据绑定

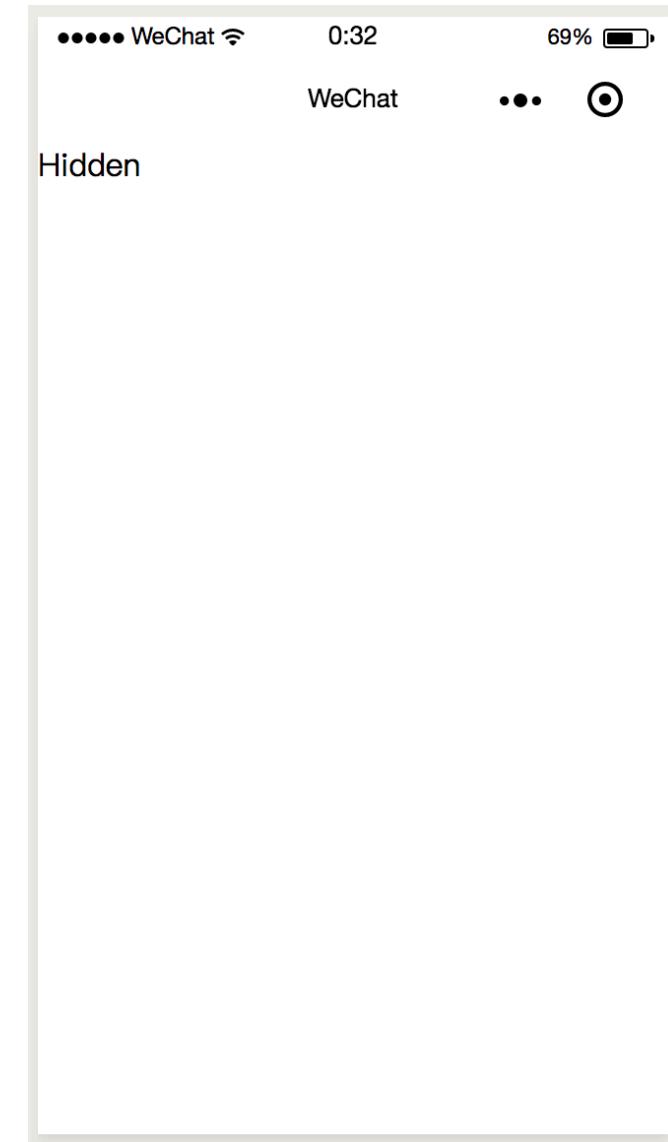
```
1  <!--index.wxml-->
2  <view>
3      <text data-name="{{theName}}>
4          </text>
5  </view>
6 // index.js
7 Page({
8     data: {
9         theName: "Jack"
10    }
11 })
```



```
<page>
  <view>
    <text data-name="Jack"></text>
  </view>
</page>
```

# 微信小程序开发框架——数据绑定

```
1  <!--index.wxml-->
2  <view hidden="{{flag ? true : false}}">
3      Hidden
4  </view>
5 // index.js
6 Page({
7     data: {
8         flag: false
9     }
10 })
```



# 微信小程序开发框架——属性

属性名	类型	描述	注解
id	String	组件的唯一标示	保持整个页面唯一
class	String	组件的样式类	在对应的 WXSS 中定义的样式类
style	String	组件的内联样式	可以动态设置的内联样式
hidden	Boolean	组件是否显示	所有组件默认显示
data-*	Any	自定义属性	组件上触发的事件时，会发送给事件处理函数
bind* / catch*	EventHandler	组件的事件	详见 <a href="#">事件</a>

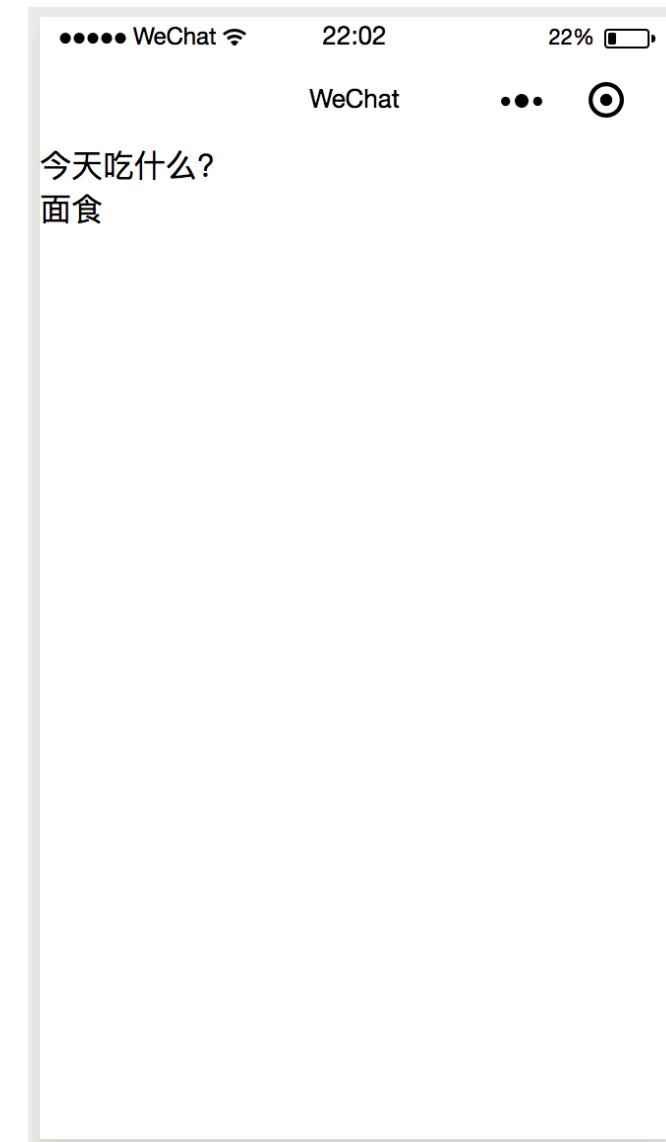
# 微信小程序开发框架——列表渲染

```
1  <!--index.wxml-->
2    <view>
3      <block wx:for="{{items}}" wx:for-item="item" wx:key="index">
4        <view>{{index}}:{{item.name}}</view>
5      </block>
6  </view>
7  // index.js
8  Page({
9    data: {
10     items: [
11       { name: "商品A" },
12       { name: "商品B" },
13       { name: "商品C" },
14       { name: "商品D" },
15       { name: "商品A" }
16     ]
17   }
18 })
```



# 微信小程序开发框架——条件渲染

```
1  <!--index.wxml-->
2  <view>今天吃什么?</view>
3  <view wx:if="{{condition === 1}}">
4  |   饺子
5  </view>
6  <view wx:elif="{{condition === 2}}">
7  |   米饭
8  </view>
9  <view wx:else>
10 |   面食
11 </view>
12 // index.js
13 Page({
14     data: {
15         condition: Math.floor(Math.random()*3+1)
16     }
17 })
18
```

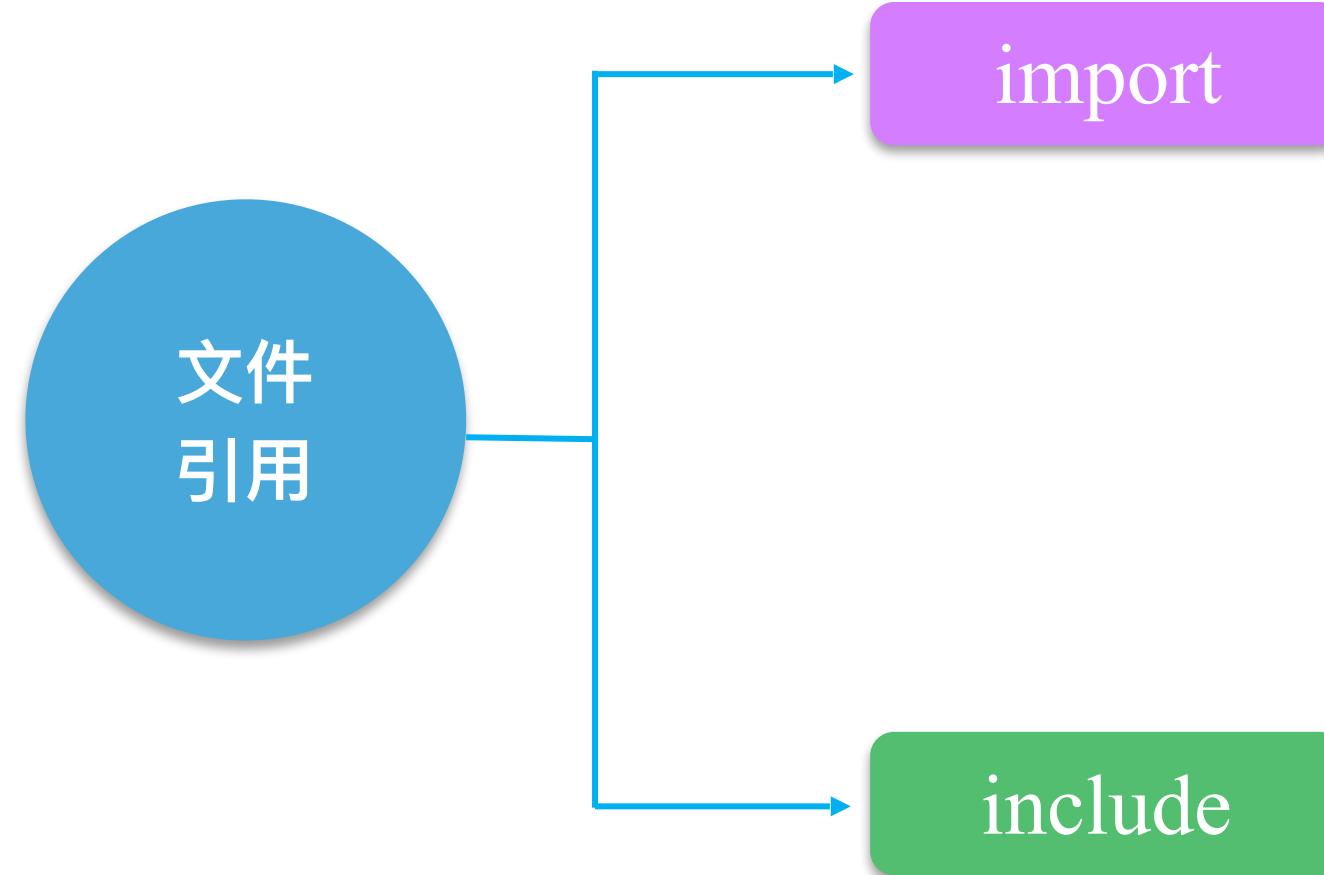


# 微信小程序开发框架——模板引用

```
1  <!--index.wxml-->
2  <template name="tempItem">
3      <view>
4          <view>收件人: {{name}}</view>
5          <view>联系方式: {{phone}}</view>
6          <view>地址: {{address}}</view>
7      </view>
8  </template>
9
10 <template is="tempItem" data="{{...item}}></template>
11 // index.js
12 Page({
13     data: {
14         item: {
15             name: "张三",
16             phone: "18888888888",
17             address: "中国"
18         }
19     }
20 })
```

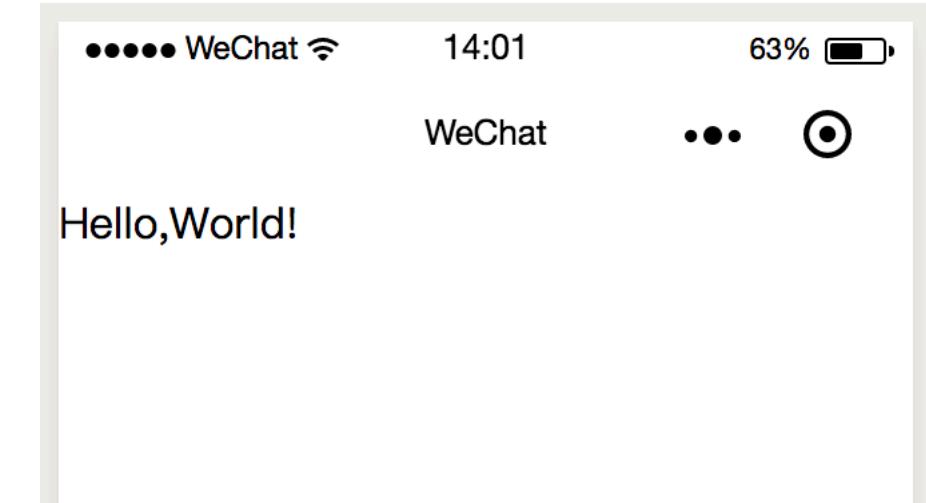


# 微信小程序开发框架——模板引用



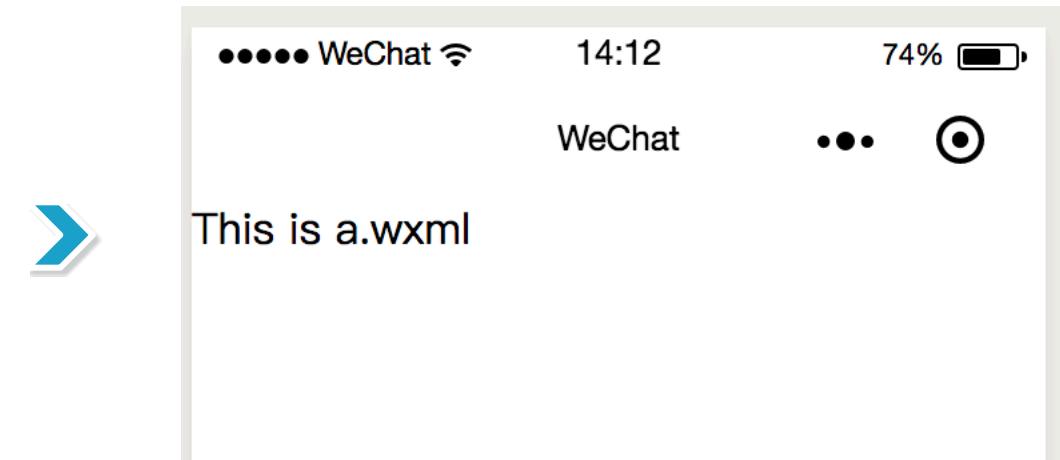
# 微信小程序开发框架——模板引用

```
1  <!--index.wxml-->
2  <import src="a.wxml"></import>
3  <template is="a"></template>
4  <!--a.wxml-->
5  <view>Hello,world</view>
6  <template name="a">
7  |   Hello,World!
8  </template>
```



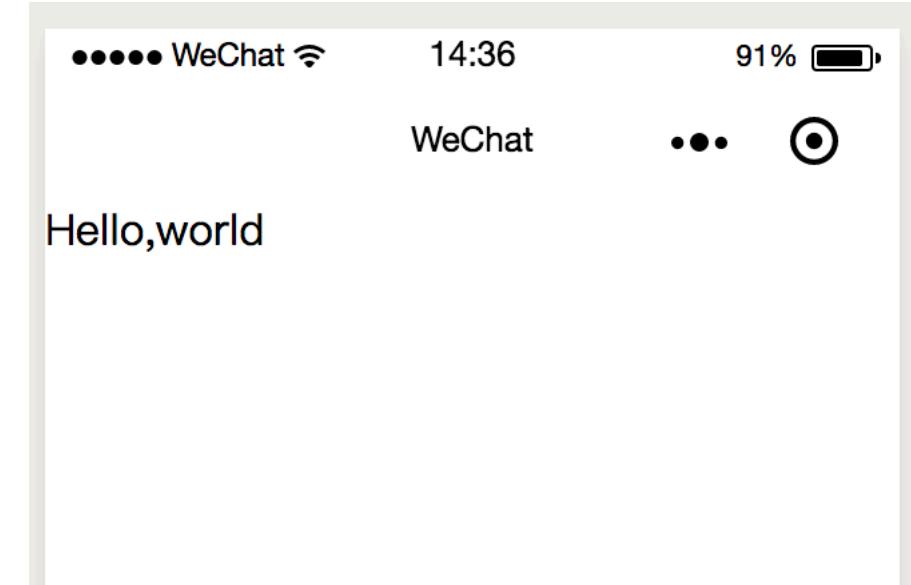
# 微信小程序开发框架——模板引用

```
1  <!--index.wxml-->
2  <import src="a.wxml"></import>
3  <template is="a"></template>
4  <!--a.wxml-->
5  <import src="c.wxml" />
6  <template name="a">
7  |   This is a.wxml
8  </template>
9  <template is="b"></template>
10 <!--b.wxml-->
11 <template name="b">
12 |   This is b.wxml
13 </template>
```



# 微信小程序开发框架——模板引用

```
1  <!--index.wxml-->
2  <include src="a.wxml" />
3  <template is="a">
4  </template>
5  <!--a.wxml-->
6  <template name="a">
7  |   <view>
8  |       This is a.wxml
9  |   </view>
10 </template>
11 <view>Hello,world</view>
```



# 微信小程序开发框架——WXSS



## WXSS (WeiXin Style Sheets)

是一套样式语言，用于描述WXML 的组件样式



## CSS (Cascading Style Sheets)

是一套样式语言，是一种样式表语言，用来描述HTML或XML文档的呈现。

width

height

position

color

border

# 微信小程序开发框架——WXSS



-  尺寸单位 `rpx`
-  样式导入
-  内联样式
-  选择器

# 微信小程序开发框架——WXSS

 设备像素 (device pixels)

 CSS像素 (CSS pixels)

 PPI/DPI (pixel per inch)

 DPR (devicePixelRatio)



# 微信小程序开发框架——WXSS

屏幕分辨率：X × Y

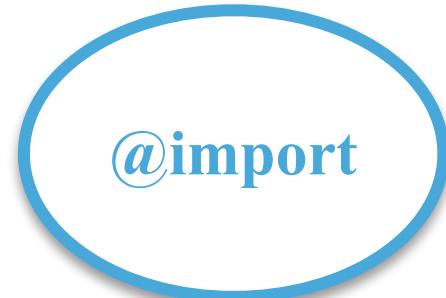
$$PPI = \frac{\sqrt{X^2 + Y^2}}{\text{屏幕尺寸}}$$



以iphone6为例

$$\frac{\sqrt{750^2 + 1334^2}}{4.7} = 325.6$$

# 微信小程序开发框架——WXSS



样式导入

执行顺序

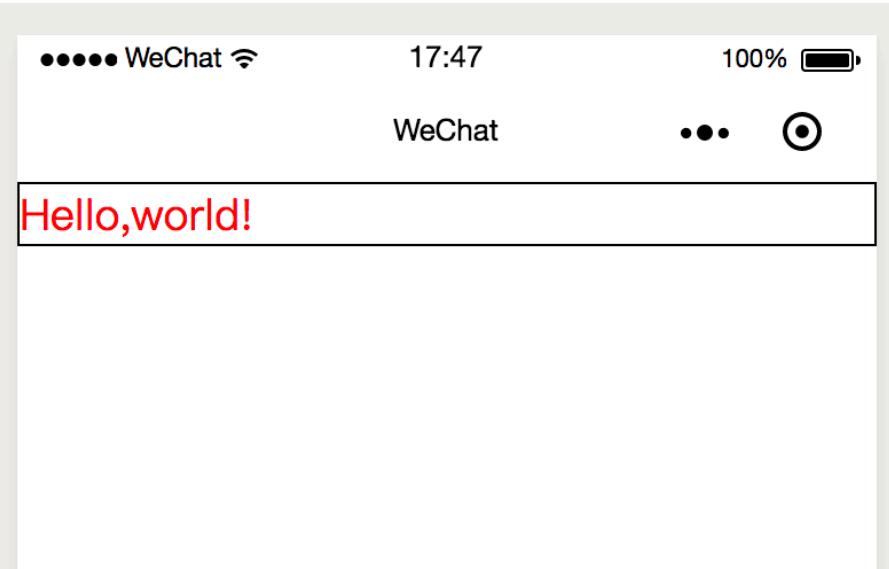


# 微信小程序开发框架——WXSS

```
1 <!--index.wxml-->
2 <view class="container">
3 |   Hello,world!
4 </view>
```

```
1 /** index.wxss */
2 @import './assets.wxss';
3 .container {
4 |   color: #red;
5 }
```

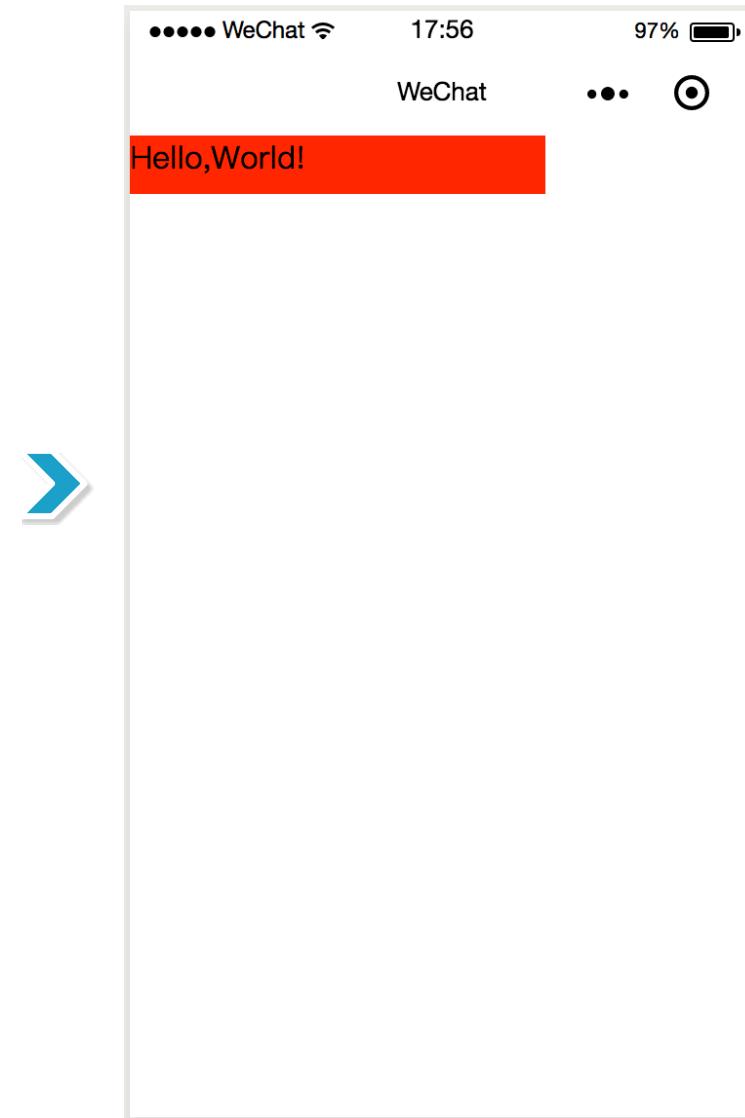
```
1 /** assets.wxss */
2 .container {
3 |   border: 1px solid #000;
4 }
```



# 微信小程序开发框架——WXSS

## 内联样式 : style

```
1 <!--index.wxml-->
2 <view style="width:500rpx;height:30px;background-color:{{colorValue}};">
3 |   Hello,world!
4 </view>
```



```
1 // index.js
2 Page({
3   data: {
4     colorValue: 'red'
5   }
6 })
```

# 微信小程序开发框架——WXSS

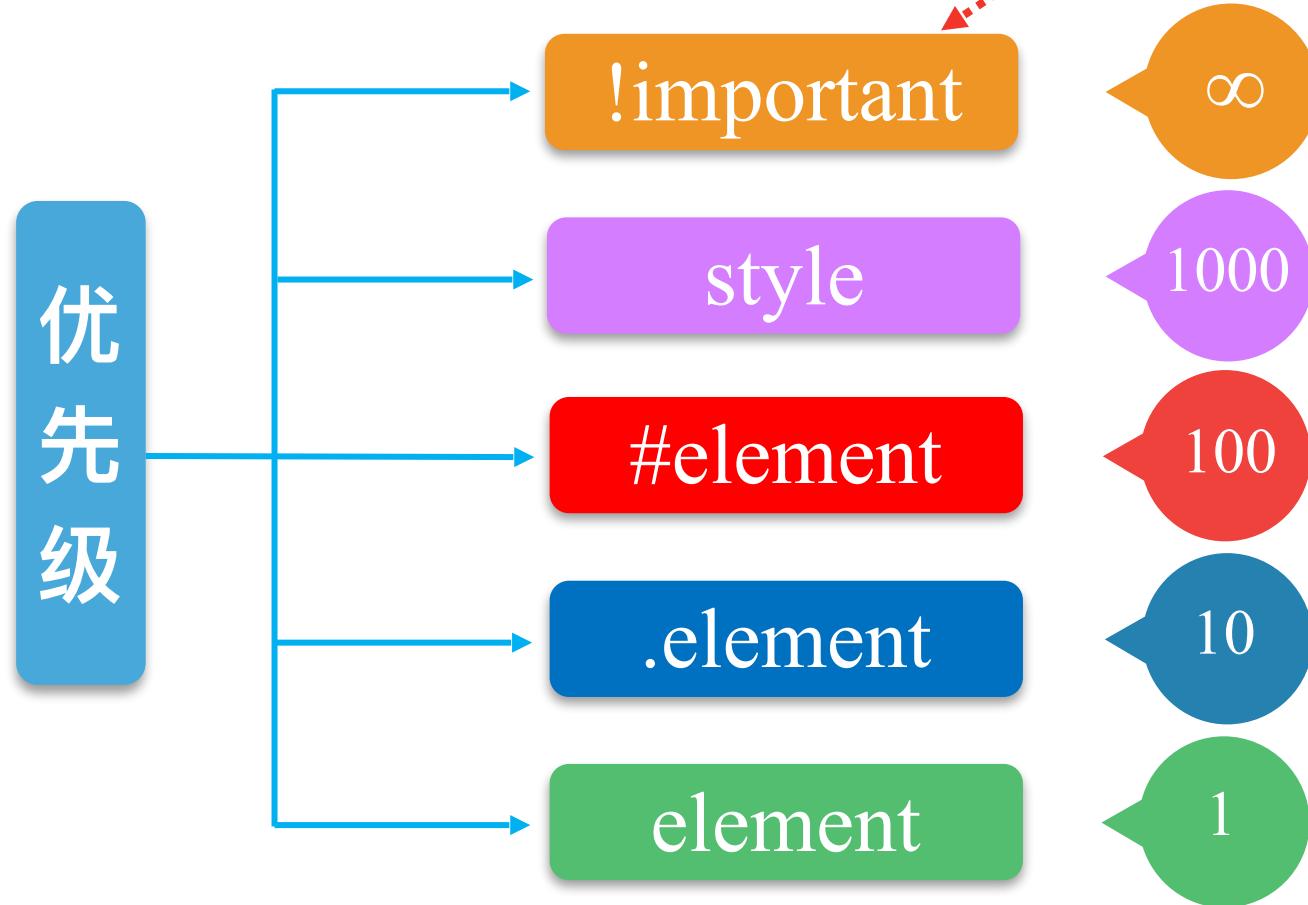
目前支持的选择器有：

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



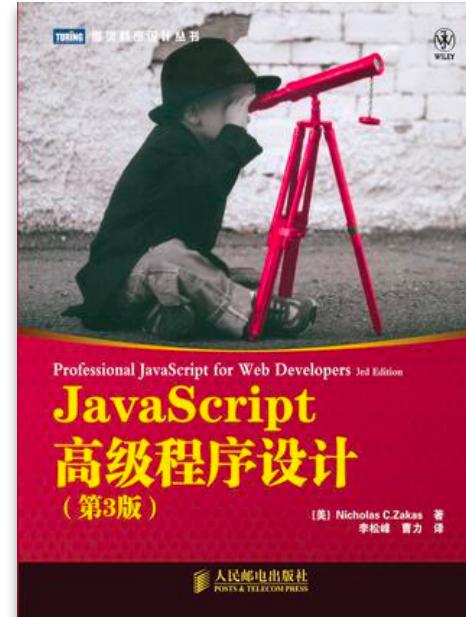
# 微信小程序开发框架——WXSS

```
1  /** index.wxss **/  
2  .title {  
3  |   color: red !important;  
4  }
```

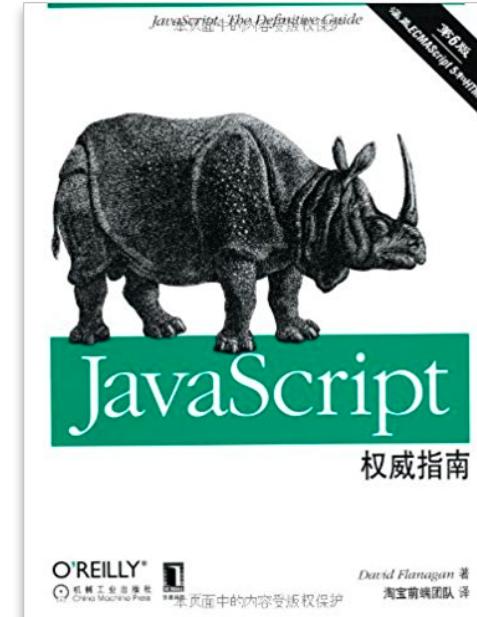


# 微信小程序开发框架——JavaScript

**JavaScript** 是一种轻量的、解释型的、面向对象的头等函数语言，是一种动态的基于原型和多范式的脚本语言，支持面向对象、命令式和函数式的编程风格。

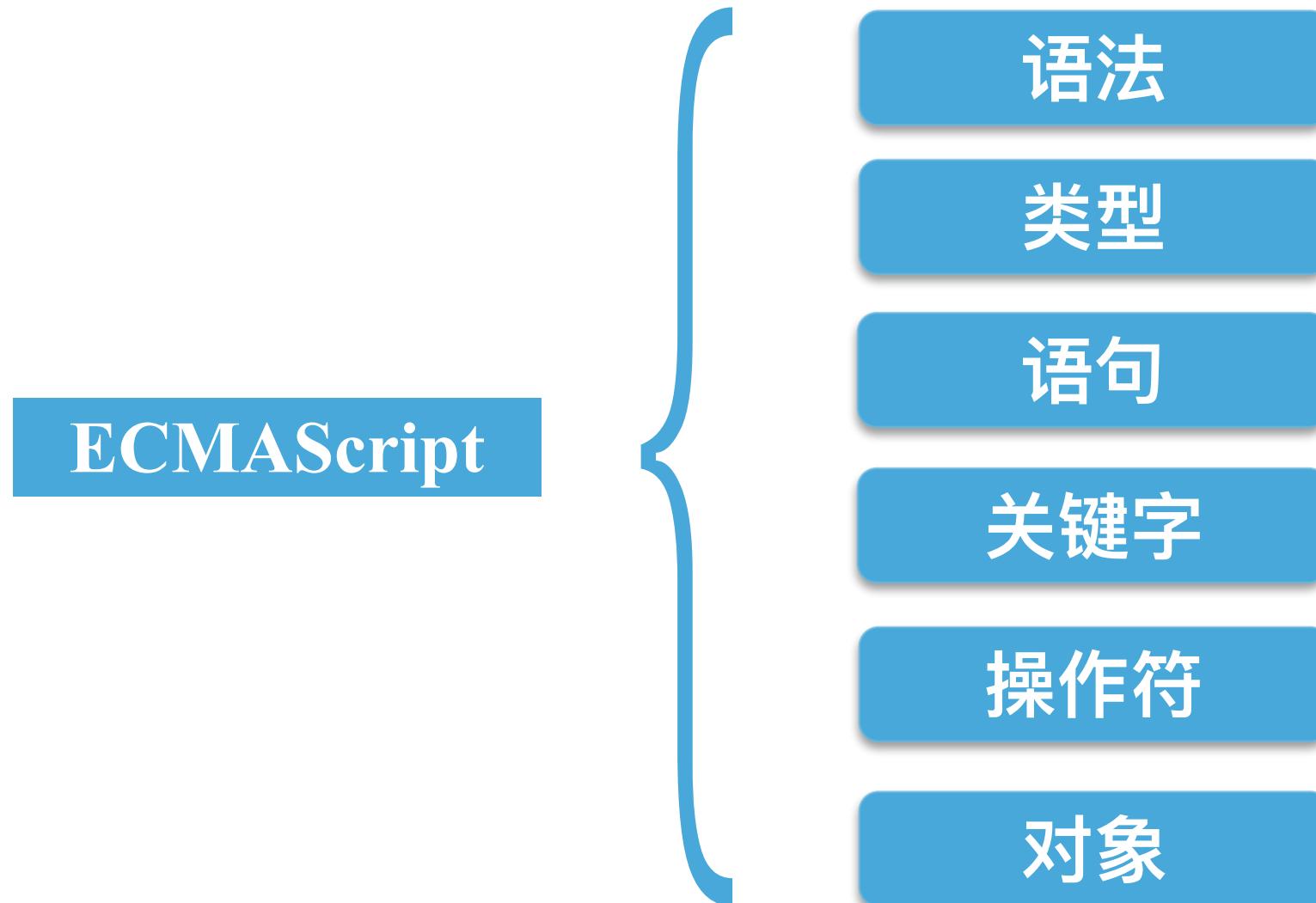


《JavaScript高级程序设计》



《JavaScript 权威指南》

# 微信小程序开发框架——JavaScript



# 微信小程序开发框架——JavaScript

## 浏览器中的JavaScript

ECMAScript

DOM

BOM

# 微信小程序开发框架——JavaScript

Nodejs中的JavaScript

ECMAScript

Native

NPM

# 微信小程序开发框架——JavaScript

小程序中的JavaScript

ECMA Script

小程序框架

小程序API

# 微信小程序开发框架——JavaScript



# 微信小程序开发框架——WXS



-  模块
-  变量
-  注释
-  运算符
-  语句
-  数据类型
-  基础类库

# 微信小程序开发框架——WXS

模块

```
<!--index.wxml-->
<wxs module="m1">
  module.exports = {
    message: 'Hello,world!'
  }
</wxs>

<view> {{m1.message}} </view>

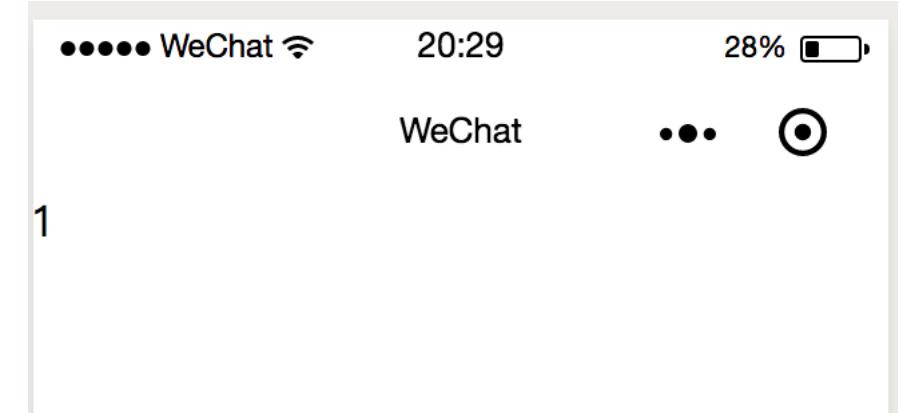
<!--index.wxml-->
<wxs src="../m2.wxs" module="m2"></wxs>
<view>{{m2.message}}</view>

// m2.wxs
module.exports = require('./m1.wxs')

// m1.wxs
module.exports = {
  message: "hello world!"
}
```

# 微信小程序开发框架——WXS

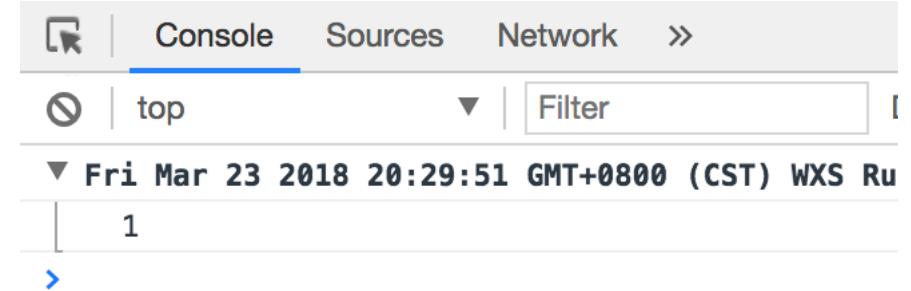
```
<!--index.wxml-->
<wxs module="m3">
    var v = 1;
    module.exports.value = v;
    // 单行注释
    /* 多行注释
    v += 1;
    */
    console.log(v);
    /*
    var d = 3;
    console.log(d);
    </wxs>
    <view>{{m3.value}}</view>
```



••••• WeChat 20:29 28% 🔋

WeChat ⌂ ⌂ ⌂ ⌂ ⌂

1



Console Sources Network >

✖ top Filter

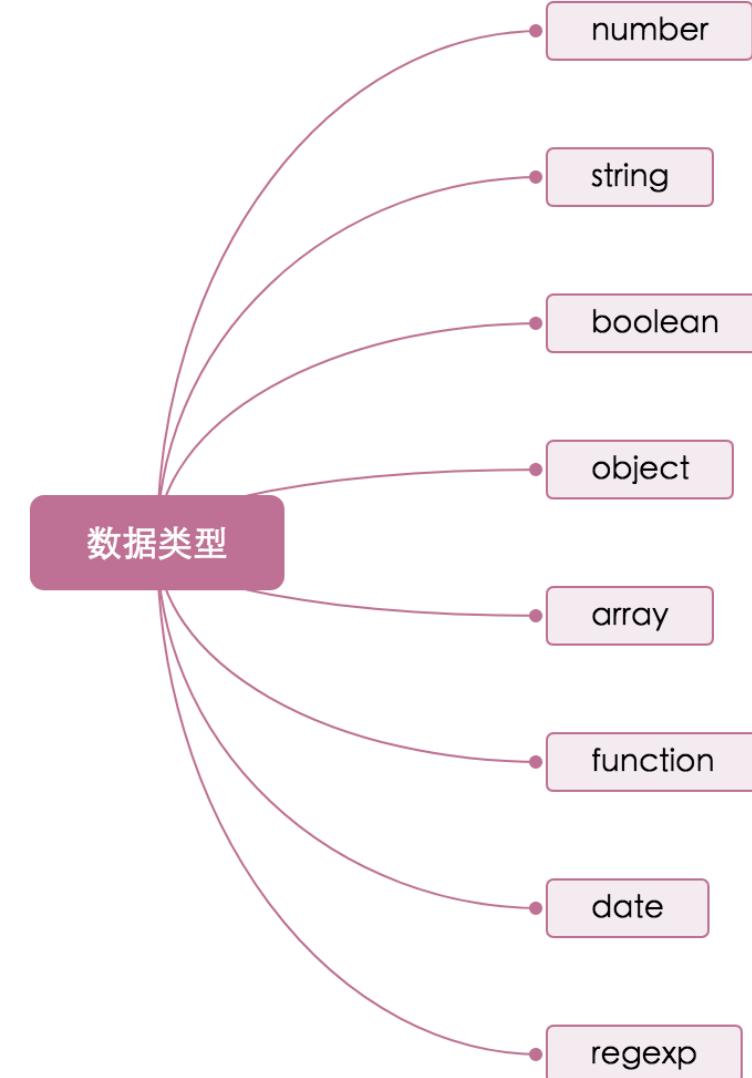
Fri Mar 23 2018 20:29:51 GMT+0800 (CST) WXS Ru

1

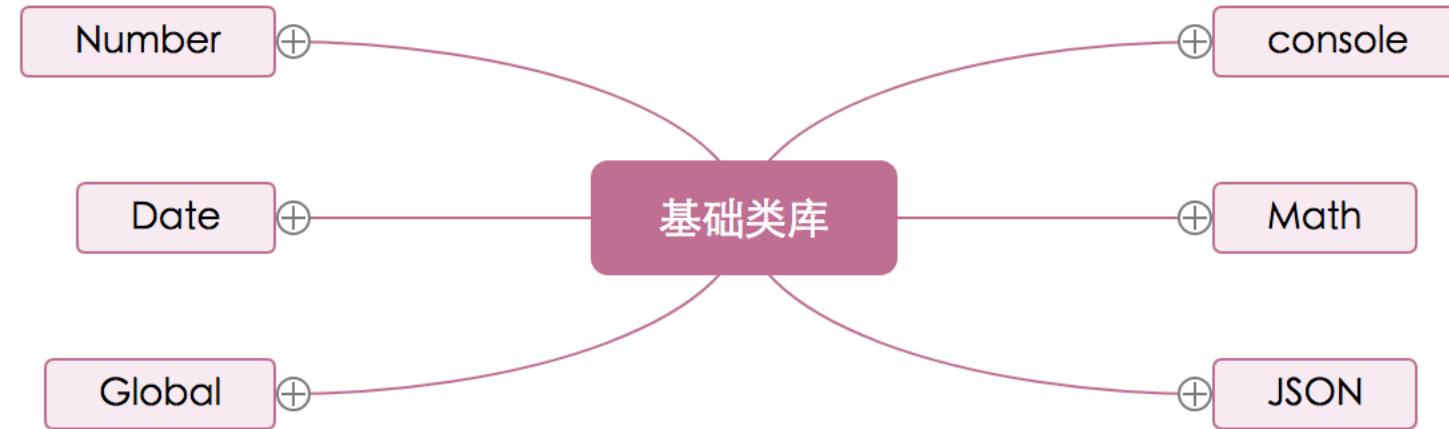
# 微信小程序开发框架——WXS



# 微信小程序开发框架——WXS



# 微信小程序开发框架——WXS

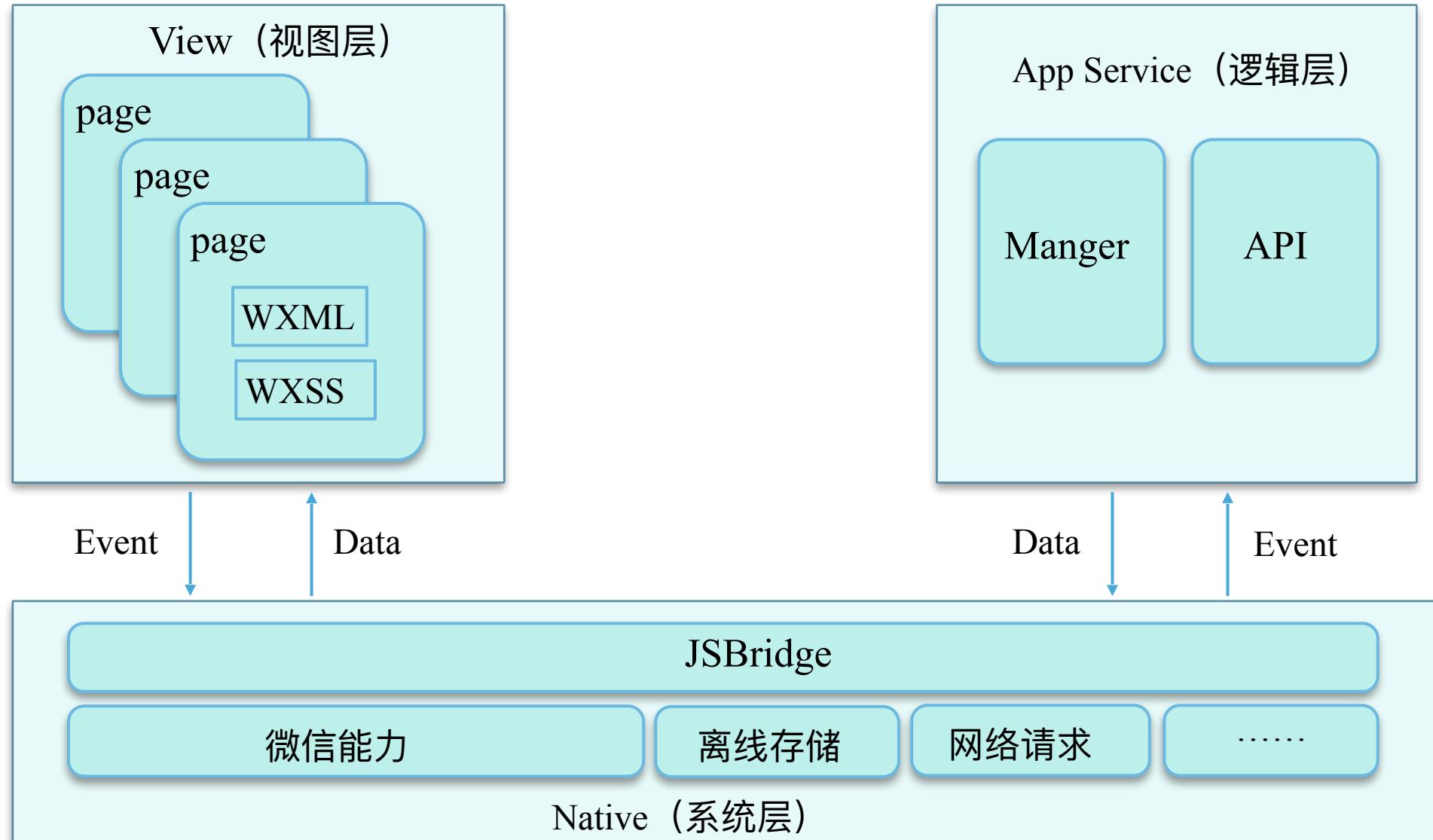


# 微信小程序开发框架——MINA框架及运行机制

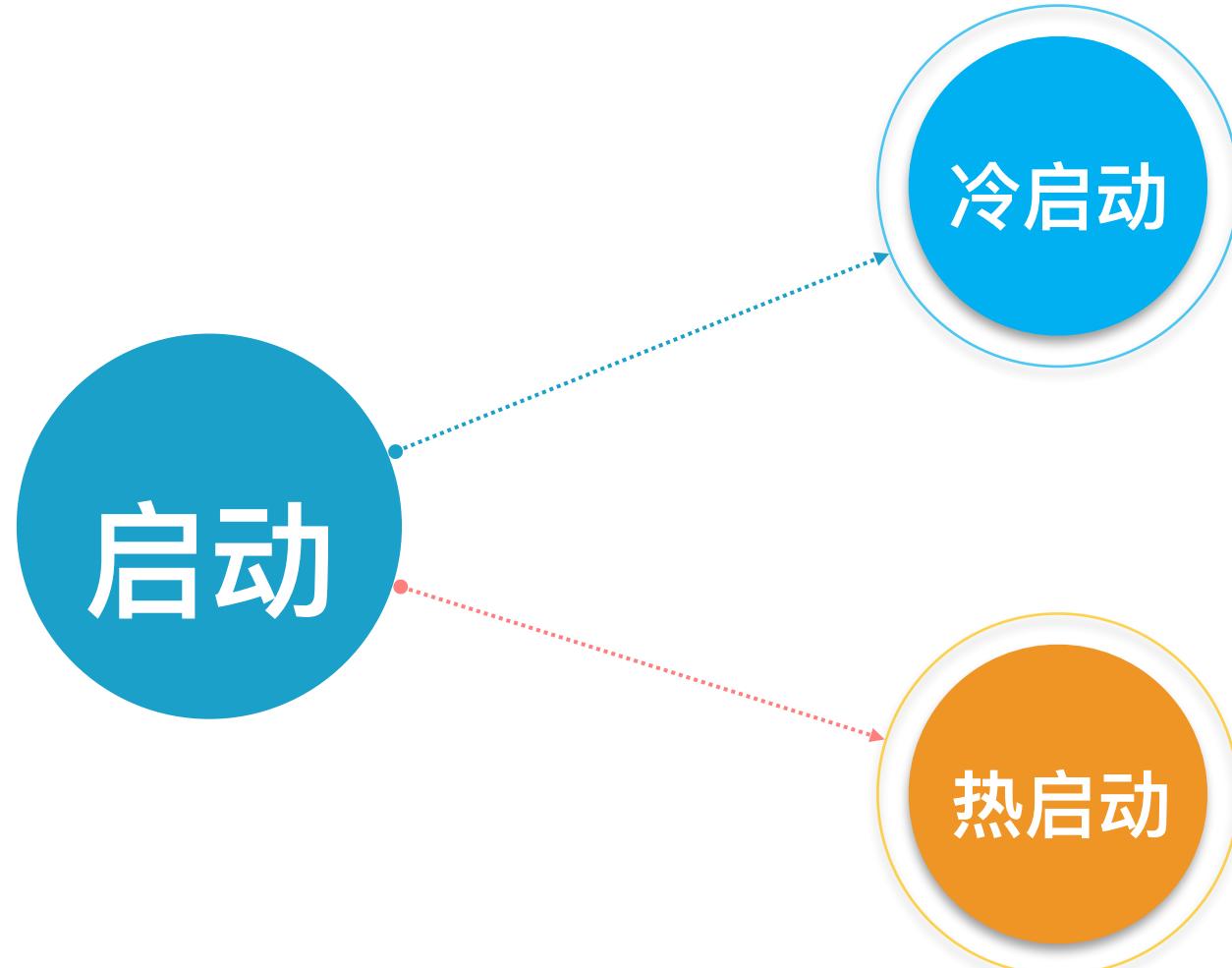
框架  
实现

运行  
机制

# MINA框架



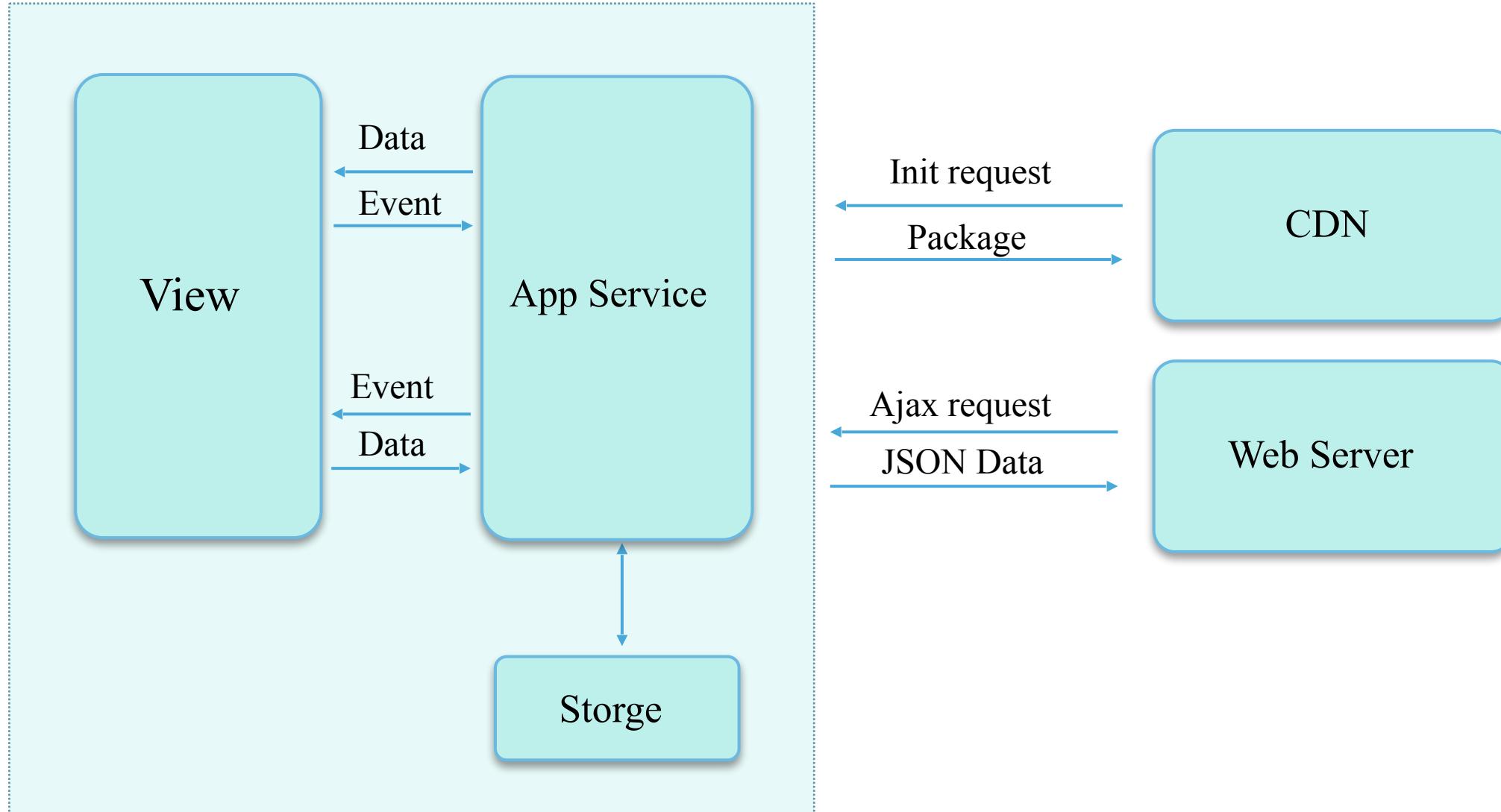
# 运行机制——启动



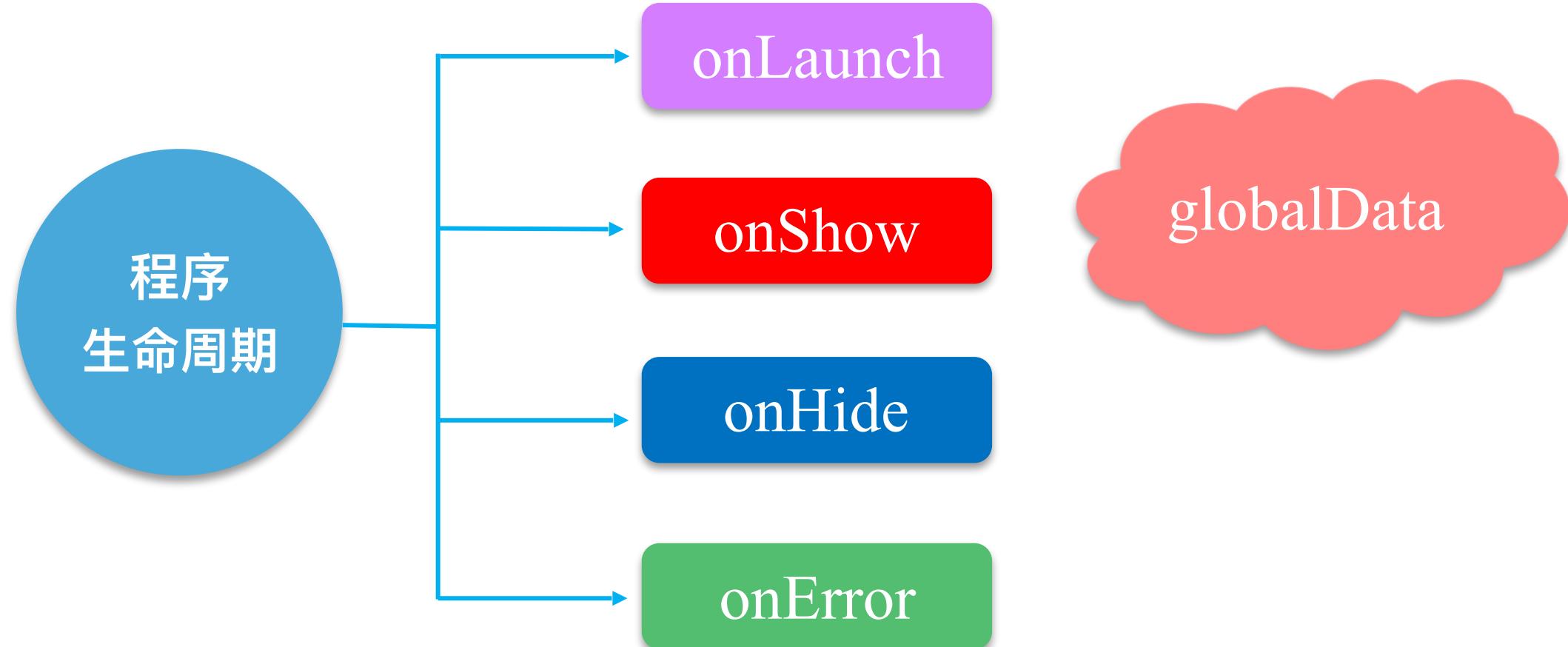
# 运行机制——加载

In WeChat

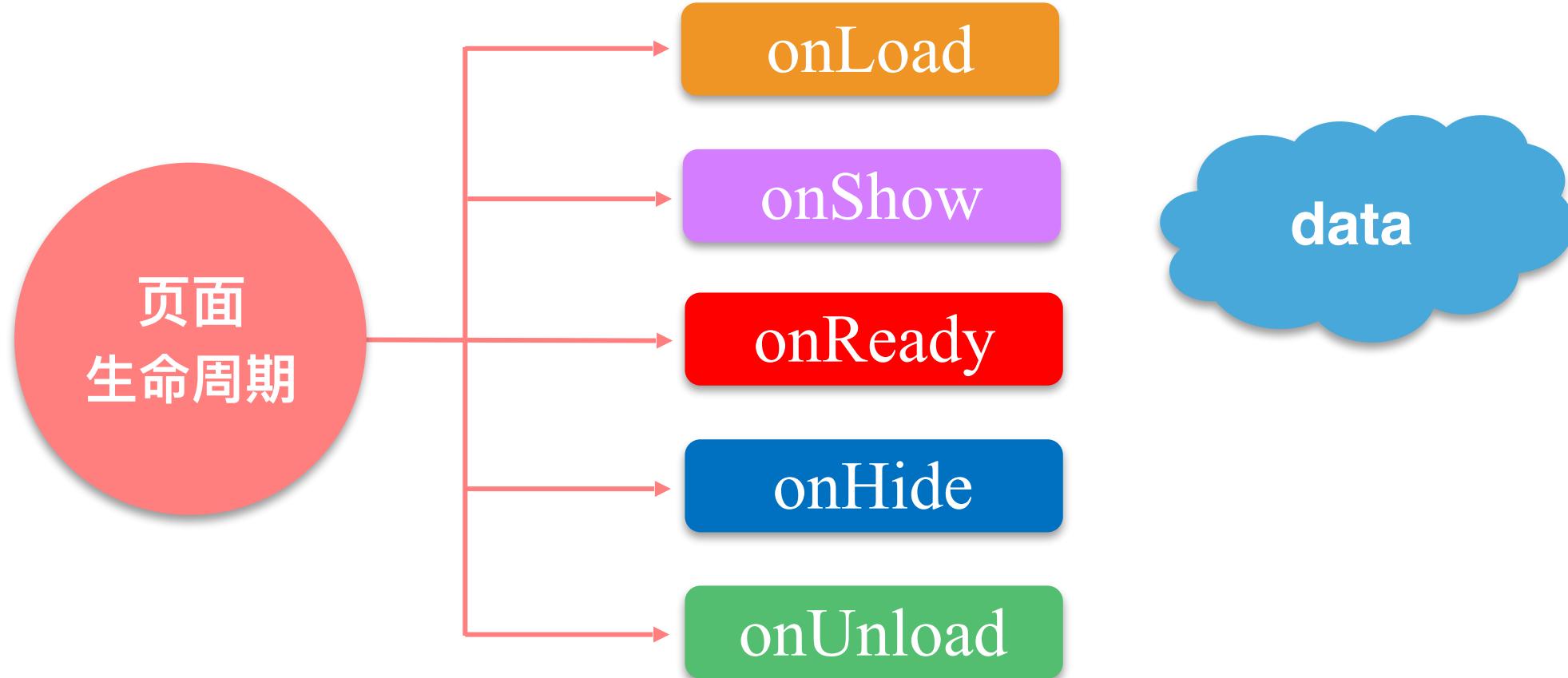
Network



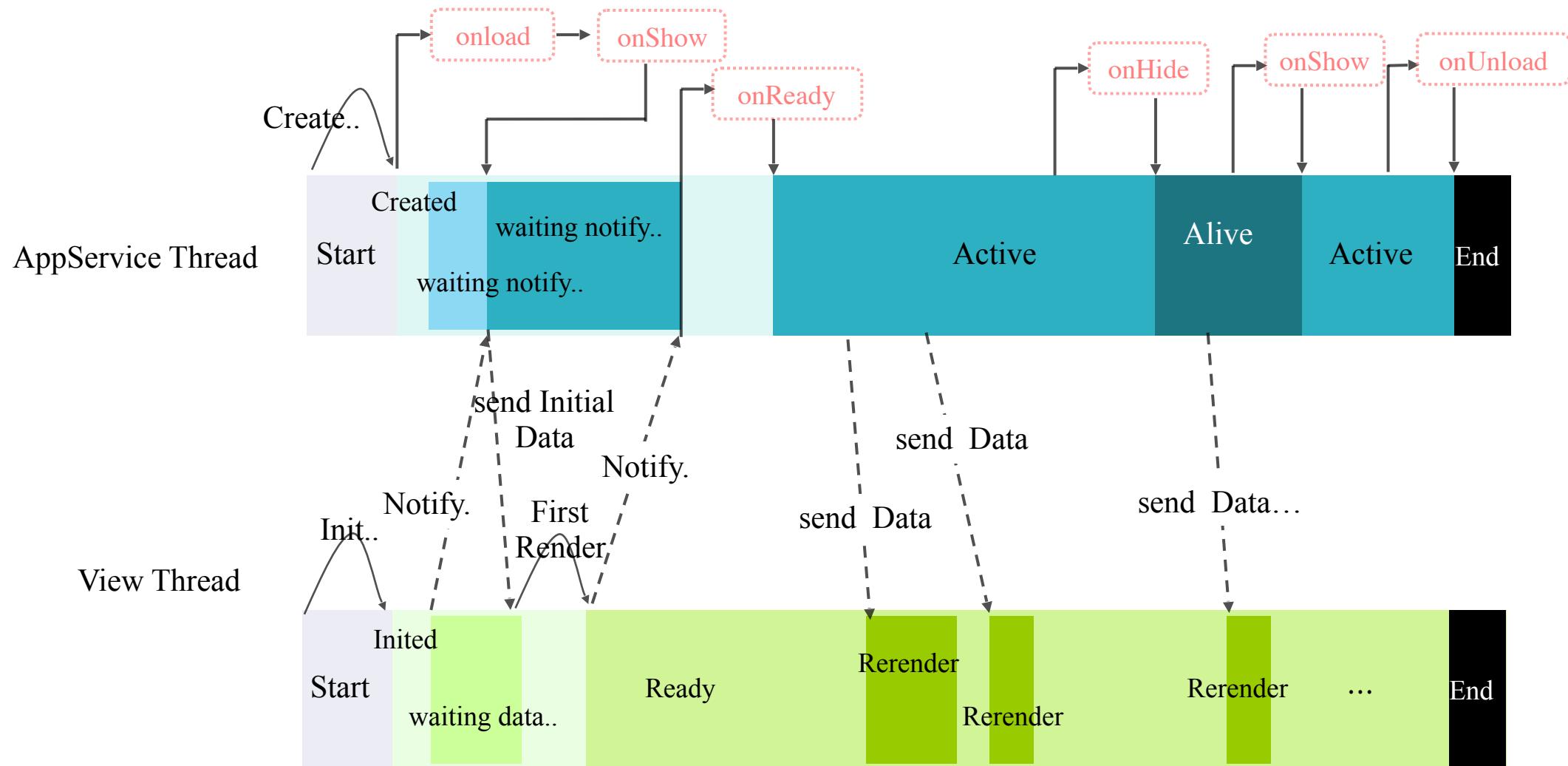
# 微信小程序开发框架——生命周期



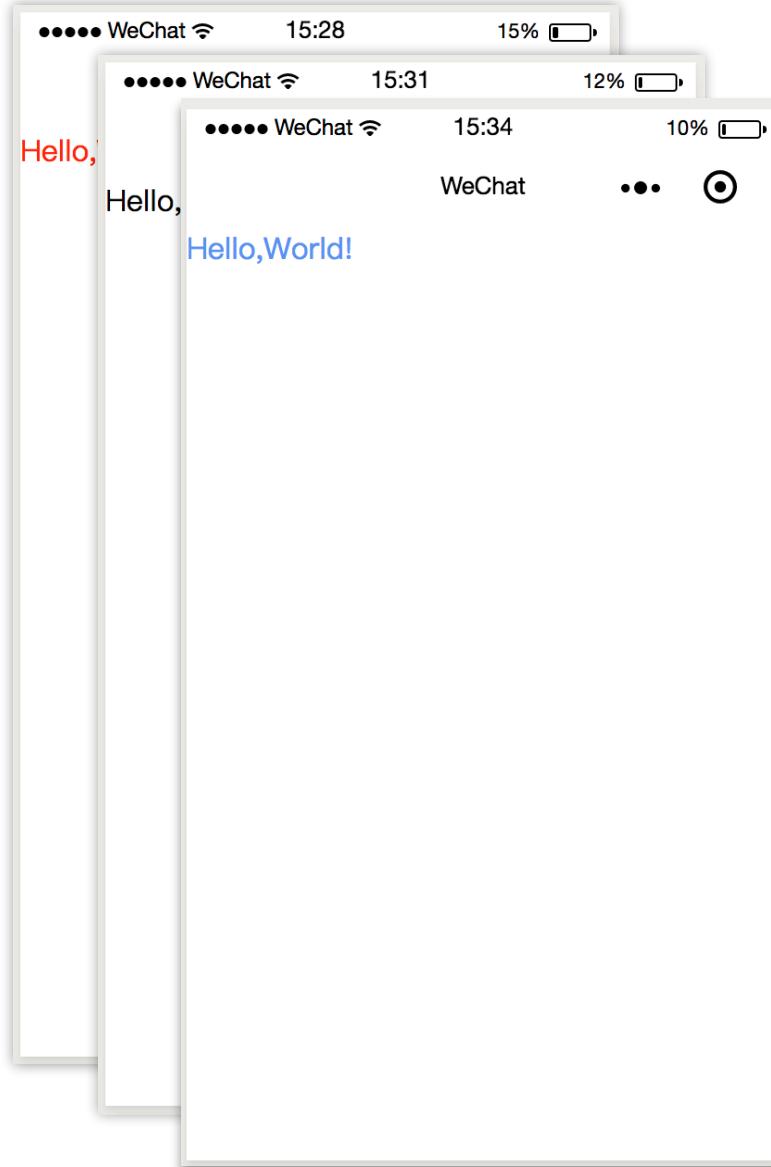
# 微信小程序开发框架——生命周期



# 微信小程序开发框架——生命周期



# 微信小程序开发框架——路由



路由方式	页面栈表现
初始化	新页面入栈
打开新页面	新页面入栈
页面重定向	当前页面出栈, 新页面入栈
页面返回	页面不断出栈, 直到目标返回页, 新页面入栈
Tab 切换	页面全部出栈, 只留下新的 Tab 页面
重加载	页面全部出栈, 只留下新的页面

# 微信小程序开发框架——路由

路由方式	触发时机	路由前页面	路由后页面
初始化	小程序打开的第一个页面		onLoad, onShow
打开新页面	调用 API <code>wx.navigateTo</code> 或使用组件 <code>&lt;navigator open-type="navigateTo"/&gt;</code>	onHide	onLoad, onShow
页面重定向	调用 API <code>wx.redirectTo</code> 或使用组件 <code>&lt;navigator open-type="redirectTo"/&gt;</code>	onUnload	onLoad, onShow
页面返回	调用 API <code>wx.navigateBack</code> 或使用组件 <code>&lt;navigator open-type="navigateBack"&gt;</code> 或用户按左上角返回按钮	onUnload	onShow
Tab 切换	调用 API <code>wx.switchTab</code> 或使用组件 <code>&lt;navigator open-type="switchTab"/&gt;</code> 或用户切换 Tab		各种情况请参考下表
重启	调用 API <code>wx.reLaunch</code> 或使用组件 <code>&lt;navigator open-type="reLaunch"/&gt;</code>	onUnload	onLoad, onShow

# 微信小程序开发框架——事件

## 事件

事件是视图层到逻辑层的通讯方式

事件可以将用户的行为反馈到逻辑层进行处理

事件可以绑定在组件上，当达到触发事件，就会执行逻辑层中对应的事件处理函数

事件对象可以携带额外信息

# 微信小程序开发框架——事件

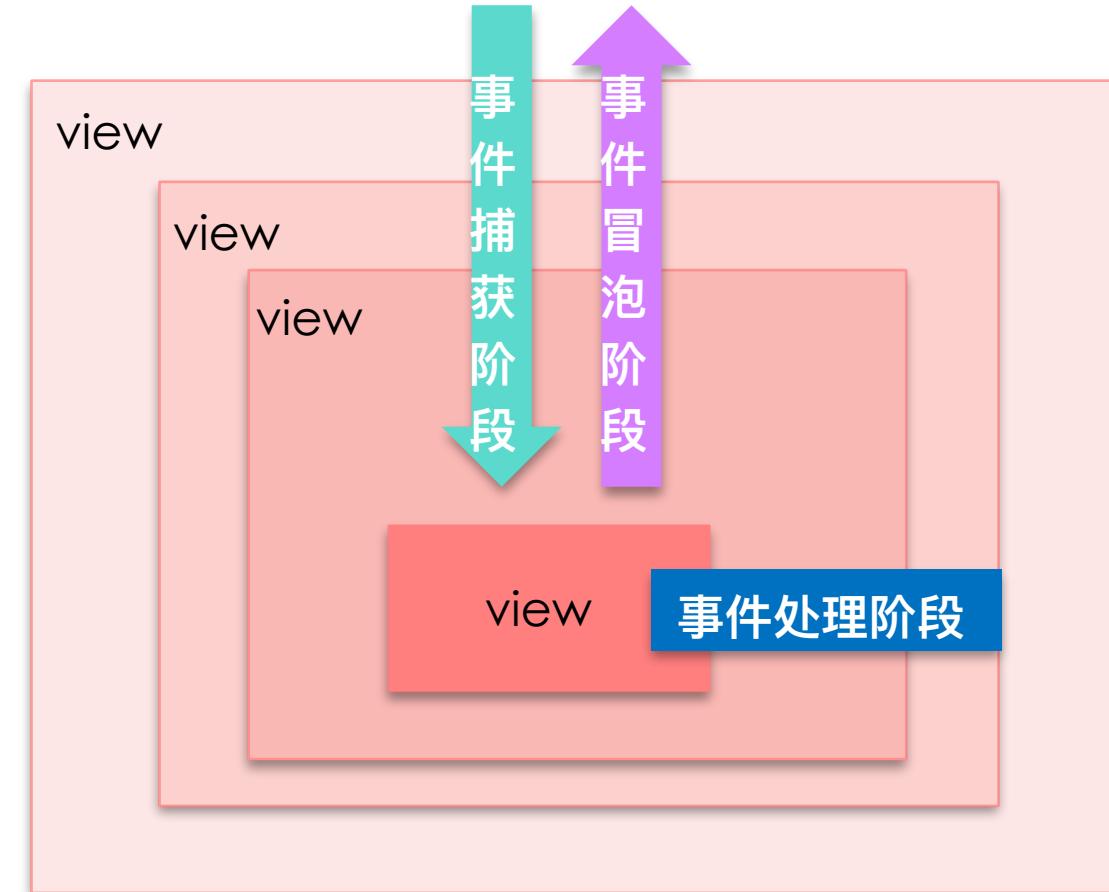
```
1  <!--index.wxml-->
2  <view>
3      <view class="btn" bindtap="clickMe">
4          点击我!
5      </view>
6  </view>
7 // index.js
8 Page({
9     clickMe(e) {
10         console.log(e);
11     }
12 })
```



```
▼ {type: "tap", timeStamp: 1169, target: {...}, currentTarget: {...}, detail: {...}, ...} ⓘ
  ► changedTouches: [...]
  ► currentTarget: {id: "", offsetLeft: 124, offsetTop: 100, dataset: ...}
  ► detail: {x: 149, y: 110}
  ► target: {id: "", offsetLeft: 124, offsetTop: 100, dataset: ...}
    timeStamp: 1169
  ► touches: [...]
  ► type: "tap"
  ► _requireActive: true
  ► __proto__: Object
```

# 微信小程序开发框架——事件

- 事件捕获阶段
- 事件处理阶段
- 事件冒泡阶段



# 微信小程序开发框架——事件

## 可捕获事件

touchstart

touchmove

touchcancel

touchend

tap

longpress

longtap

## 可冒泡事件

touchstart

longtap

touchmove

transitionend

touchcancel

animationstart

touchend

animationiteration

tap

animationend

longpress

touchforcechange

# 微信小程序开发框架——组件

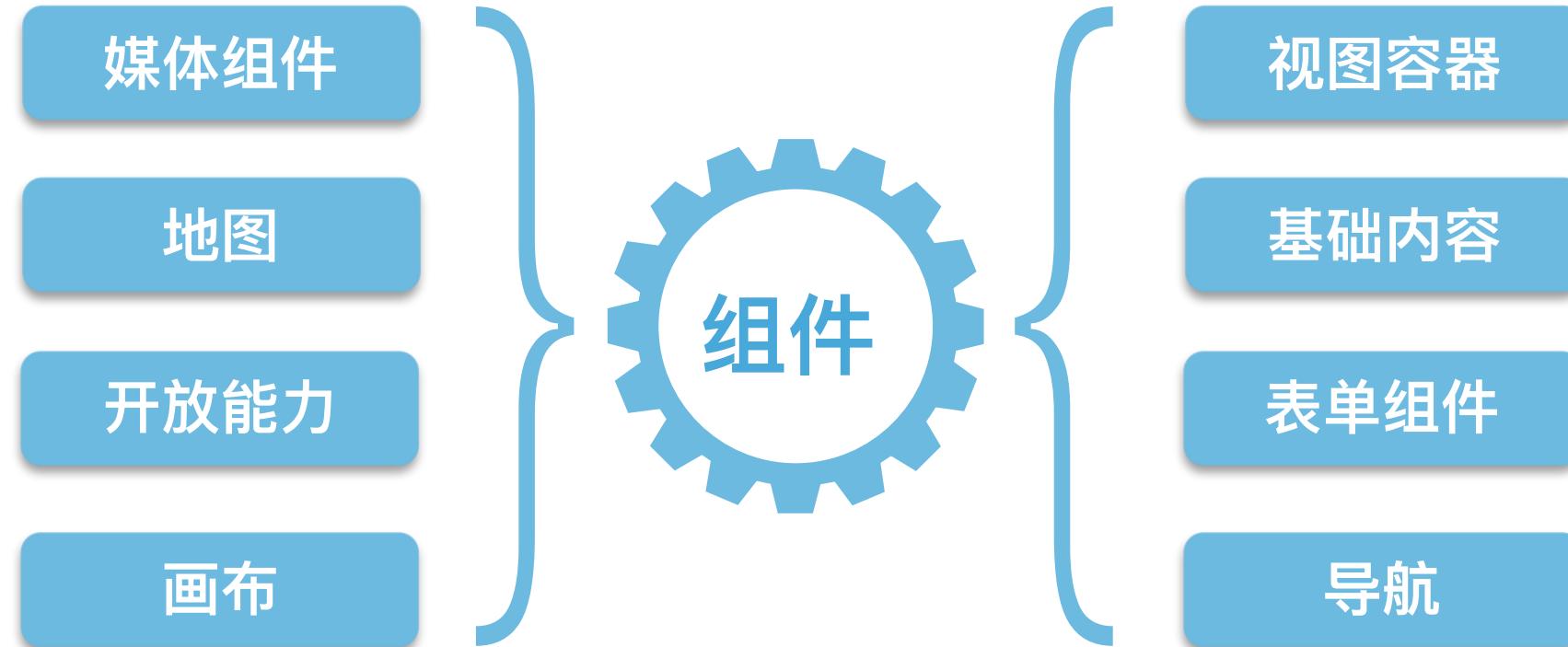


组件是视图层的  
基本组成单元

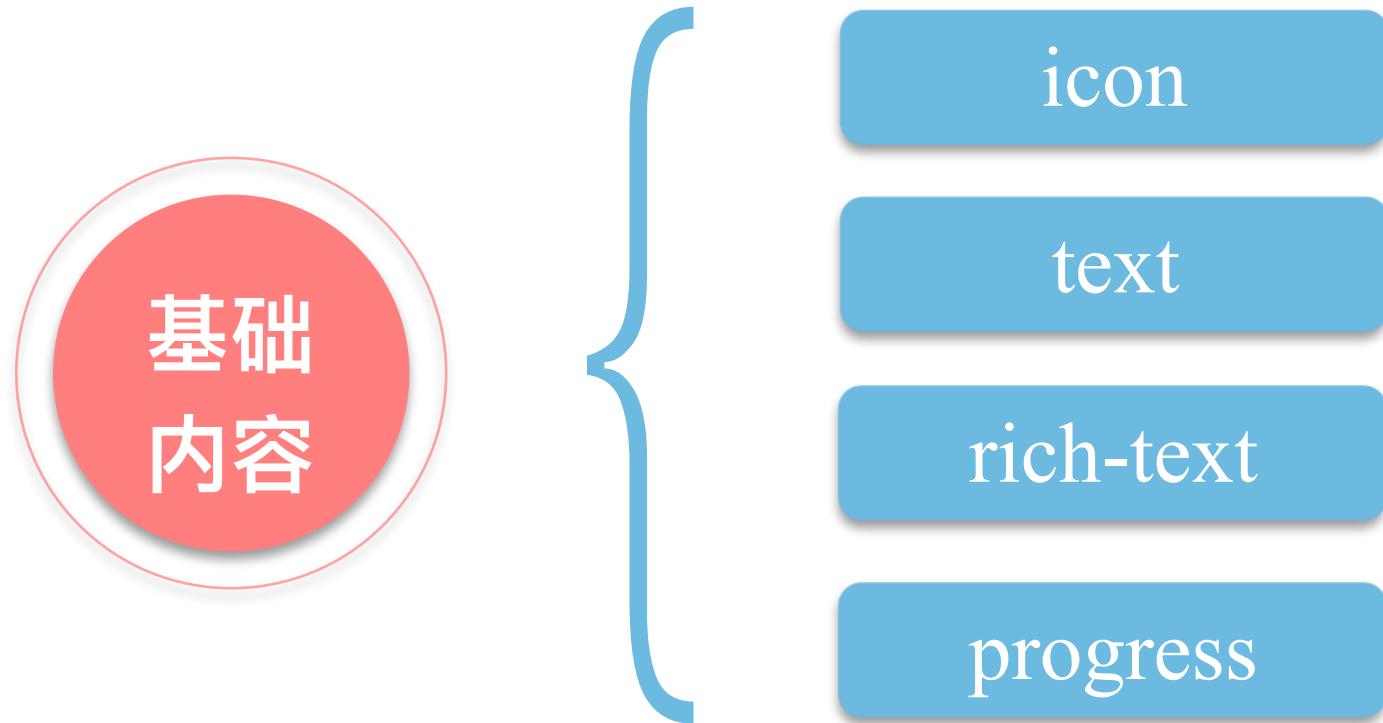
组件自带一些功能与  
微信风格的样式

一个组件通常包括:开始标签和结束标签  
属性用来修饰这个组件  
内容在两个标签之内。

# 微信小程序开发框架——组件



# 微信小程序开发框架——组件



# 微信小程序开发框架——组件

view

movable-view

swiper

cover-view

scroll-view

视图容器

# 微信小程序开发框架——组件

表单  
组件

button

picker

checkbox

picker-view

form

radio

input

switch

label

text-area

# 微信小程序开发框架——组件



导航组件



navigato

# 微信小程序开发框架——组件



audio

image

video

live-player

camera

live-pusher

# 微信小程序开发框架——组件



地图组件

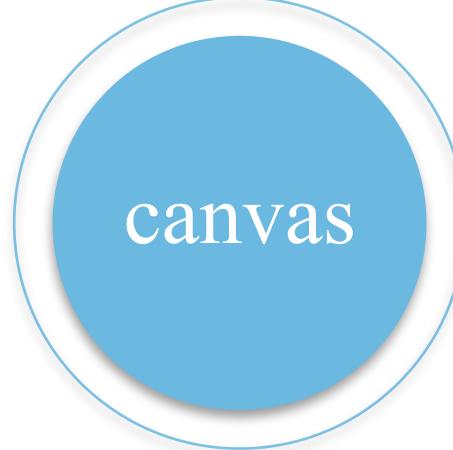


map

# 微信小程序开发框架——组件



画布组件



canvas

# 微信小程序开发框架——组件



open-data

web-view

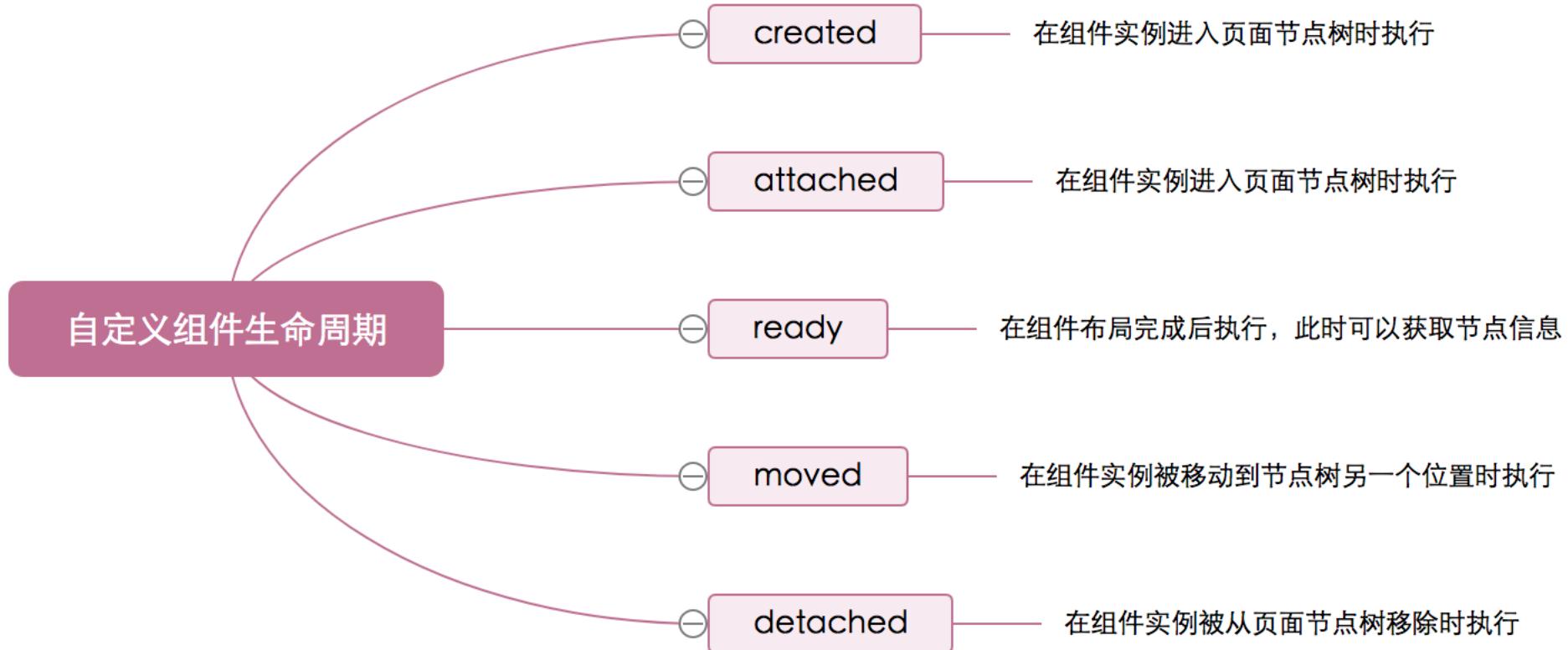
# 微信小程序开发框架——组件



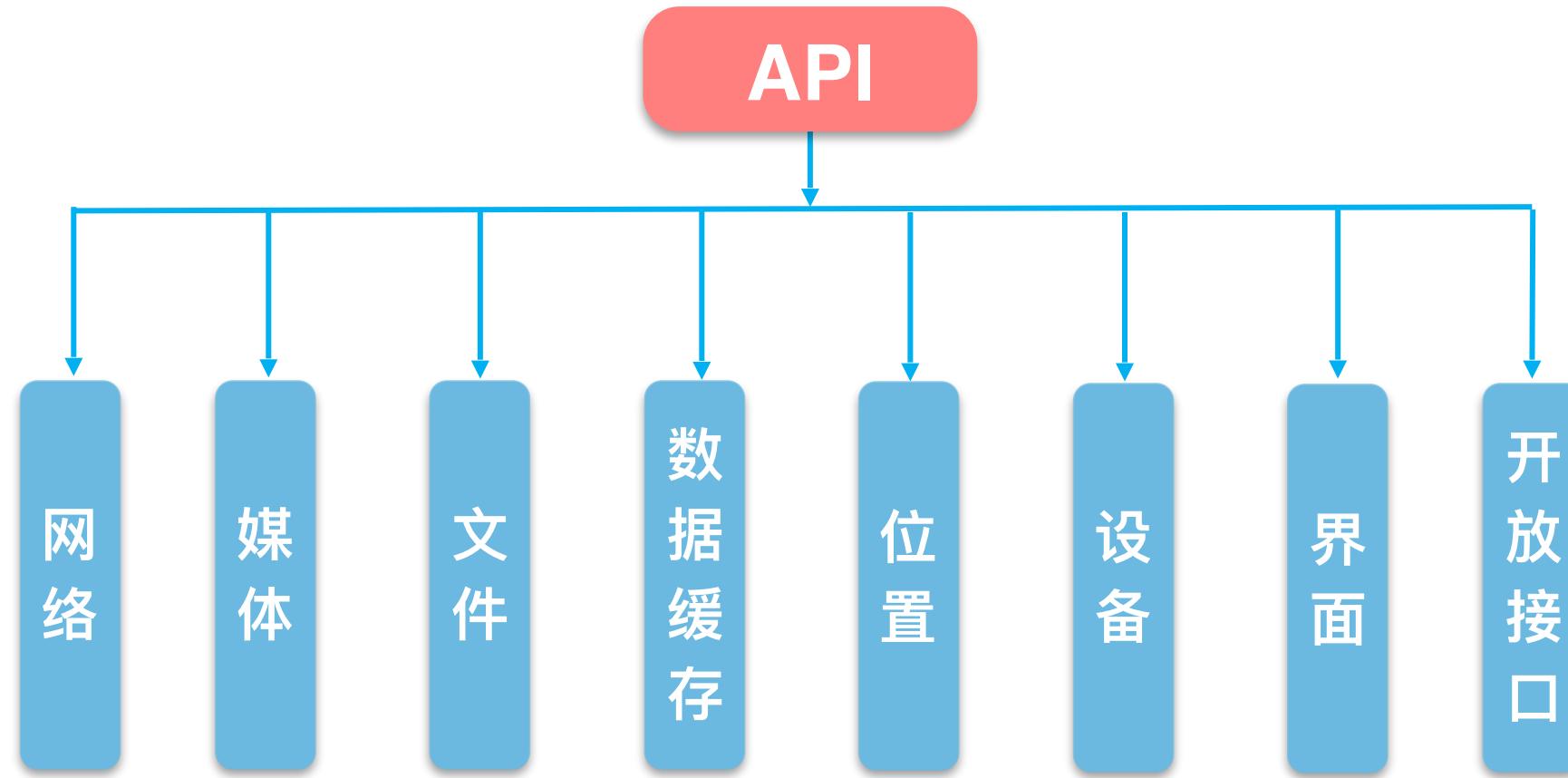
自定义组件 ?

怎么写自定义组件 ?

# 微信小程序开发框架——组件



# 微信小程序开发框架——API



# 微信小程序开发框架——API



-  **wx.on**
-  **Object参数**
-  **wx.get/wx.set**

success

fail

complete

# 微信小程序开发框架——API

直调函数

回调函数

# 微信小程序开发框架——API

HTTP

是基于TCP/IP通信协议通过万维网服务器传输  
数据到本地浏览器的应用层协议

websocket

是由HTML5规范提出的一种在单个TCP连接上  
进行全双工通信的应用层协议

# 微信小程序开发框架——API

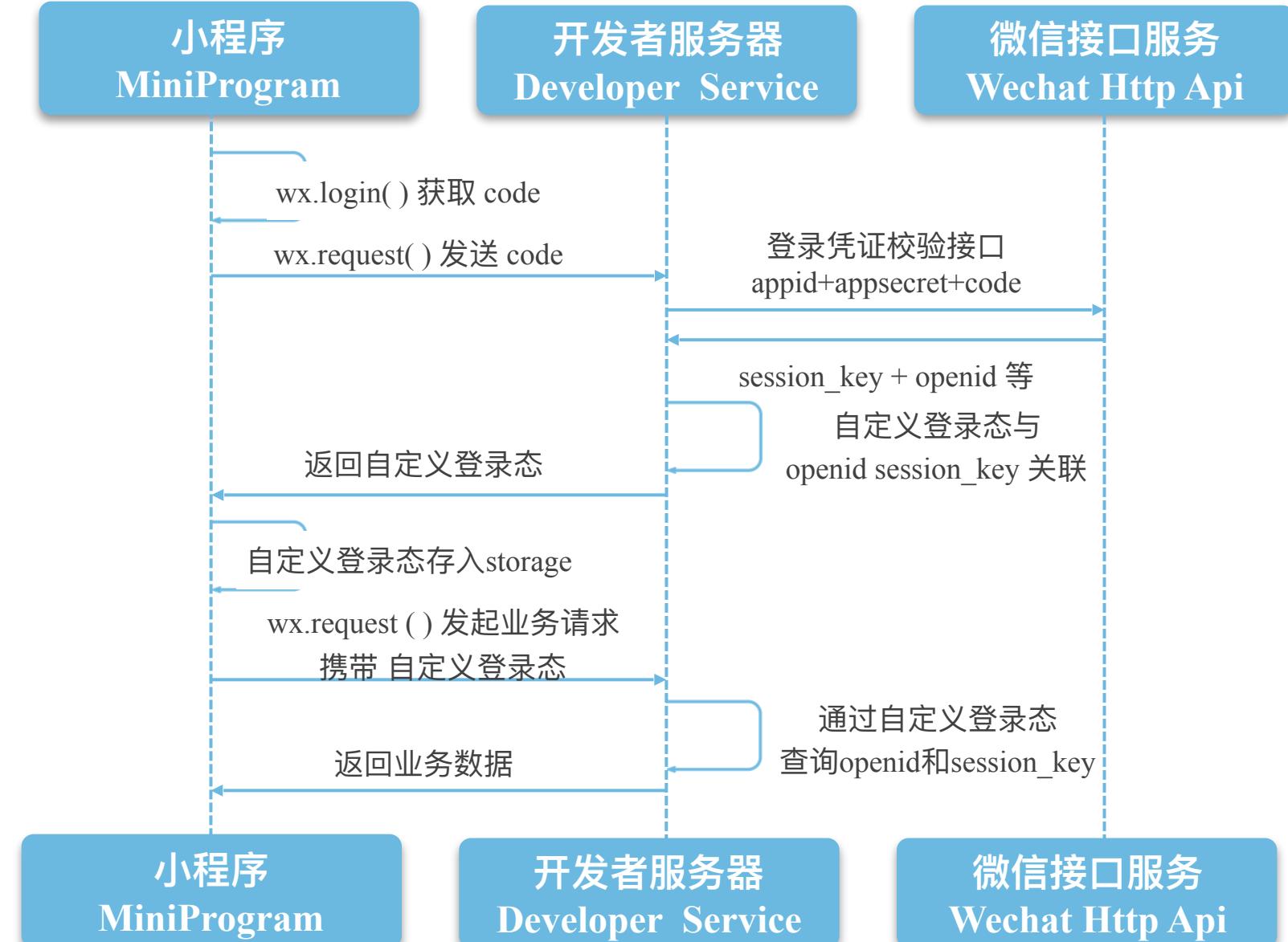


微信 APP iOS : 6.5.21

微信 APP android : 6.5.19

小程序基础库最低版本要求: 1.7.0

# 微信小程序开发框架——登录授权



# 微信小程序开发框架——登录授权

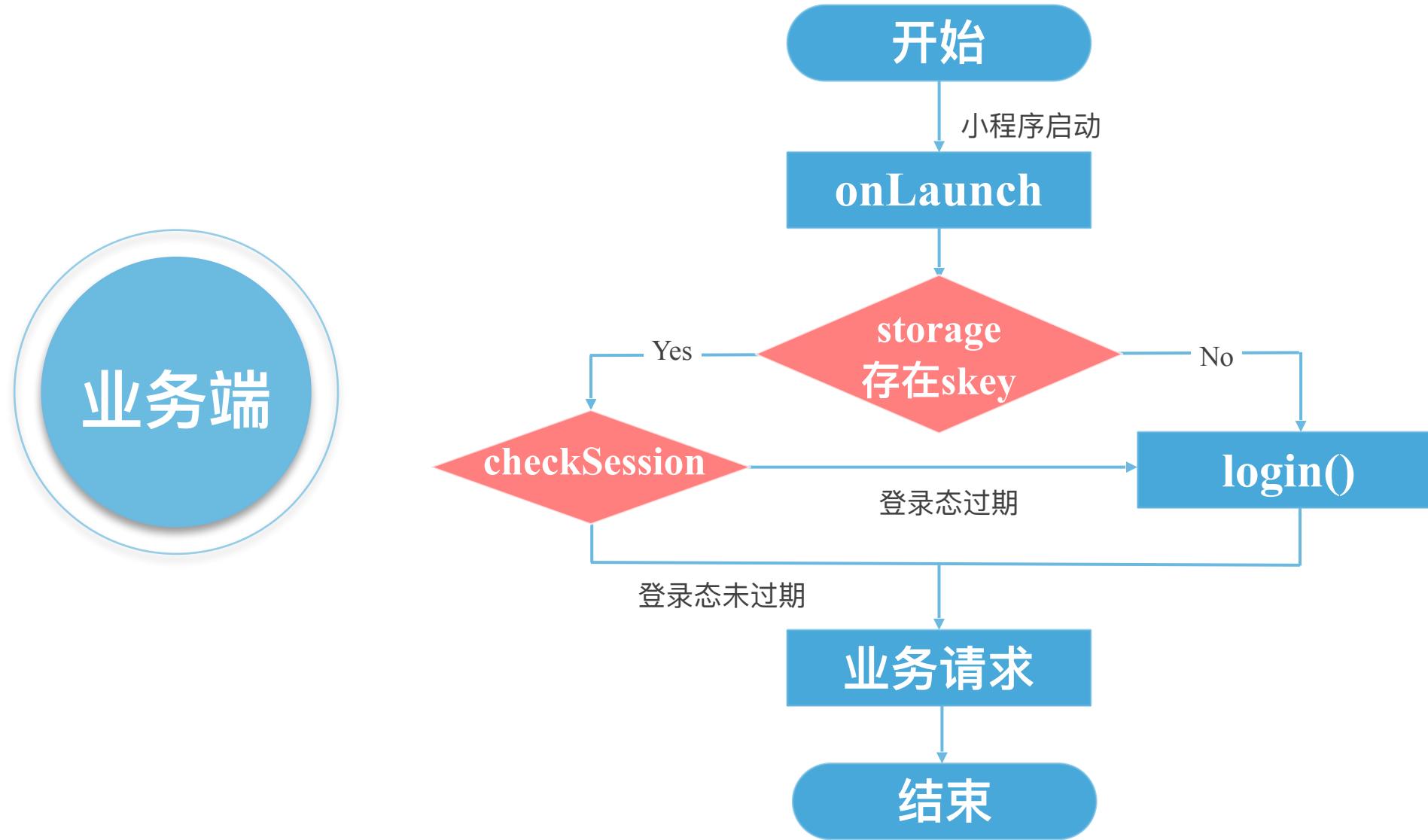
## 注意事项

code和session\_key  
为一一对应的关系

session\_key有时效性



# 微信小程序开发框架——登录授权



# 微信小程序核心功能——微信支付



微信公众平台 | 小程序

文档 社区 帮助

首页 开发管理 用户身份 数据分析 模板消息 客服消息 附近的小程序 微信支付 推广 设置

## 微信支付

帮助 ?

 微信支付  
未开通

**开通**

**申请条件**  
申请微信小程序支付，必需满足以下条件：  
• 通过[微信认证](#) 已认证

**介绍**  
微信小程序支付，是微信向有出售物品需求的小程序提供的支付收款、经营分析的整套解决方案。  
[申请指引](#) | [开发文档](#)

**申明**  
申请微信支付，包括微信认证已提供的资料外，你还需要提供以下材料：  
• 联系人姓名、手机号码、常用邮箱  
• 公司网址(非互联网公司可不填)  
• 商家名称(将用于对外展示)  
• 售卖商品描述(140字以内)  
• 客服电话



## 微信支付 | 商户平台 开通微信支付

选择开通方式

**申请一个新的微信支付商户号**

**新申请**

流程 (预计耗时: 1-5天)  
填写申请信息  
经过微信支付审核  
查收开户邮件  
验证打款金额  
登录商户平台签署协议

**绑定已有微信支付商户号**

**绑定**

流程 (预计耗时: 10分钟)  
验证原有商户号帐密信息  
判断是否符合绑定条件  
绑定成功后即可进行微信支付开发

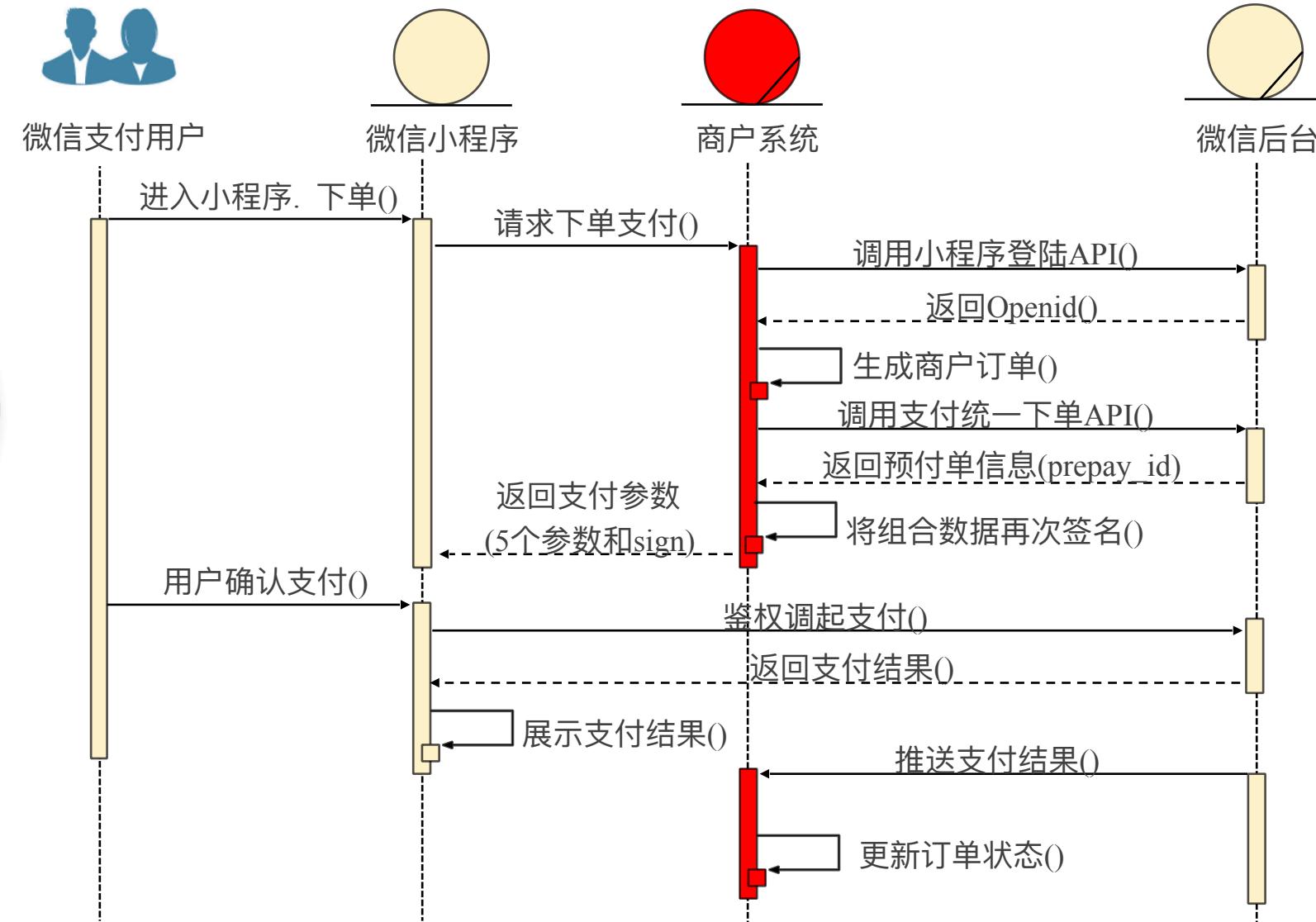
通过以上两种方式，均可使你的APPID获取微信支付的能力。  
两种开通方式均不可逆，请根据自身业务情况选择开通方式。

# 微信小程序核心功能——微信支付

设置密钥并  
下载商户证书

确认web服务  
器https服务

# 微信小程序核心功能——微信支付



绘制交互

# 微信小程序核心功能——模板消息



推送位置 服务通知

下发条件 支付/提交表单

跳转能力 小程序各个页面

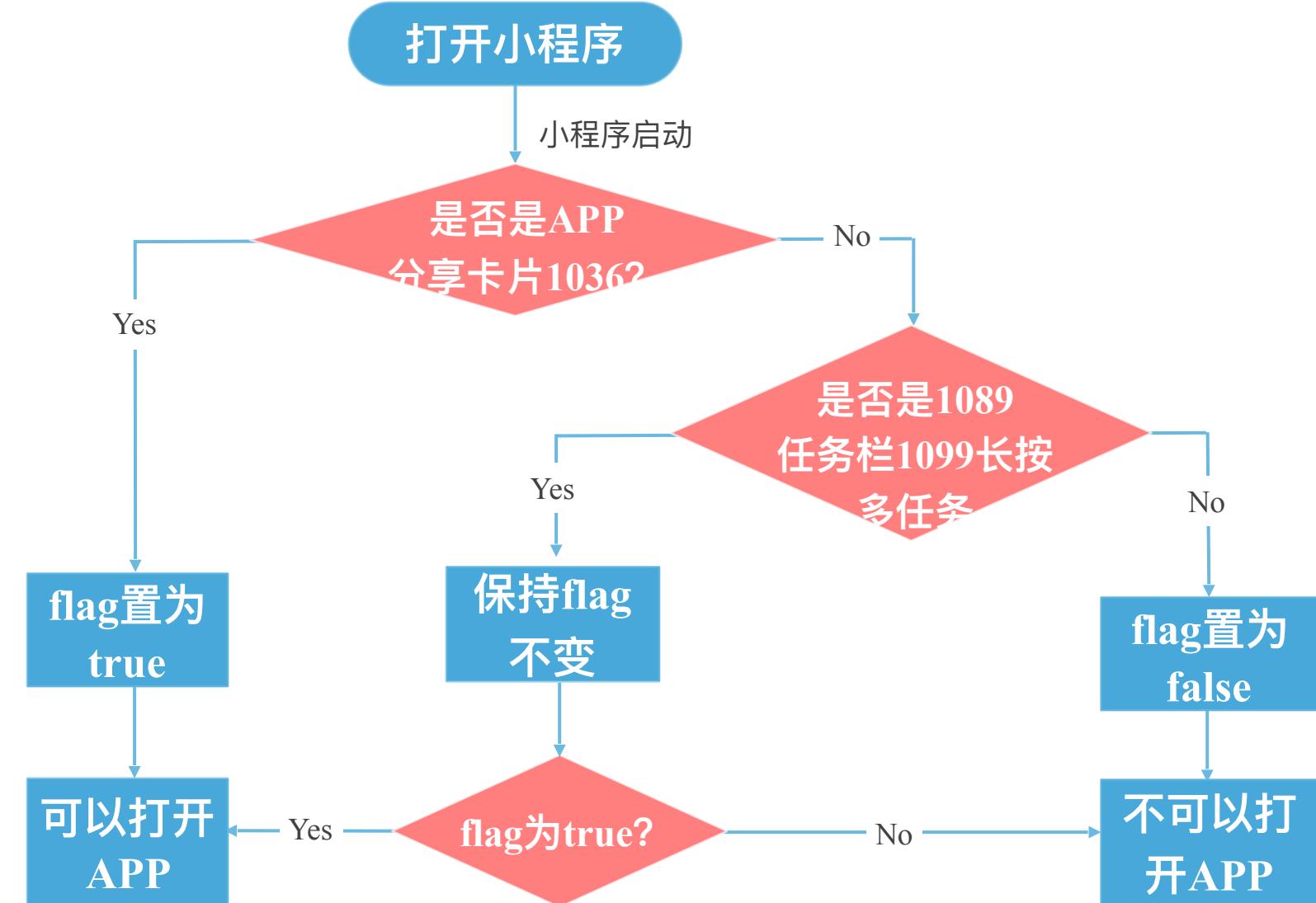
# 微信小程序核心功能——转发

Page.onShareAppMessage

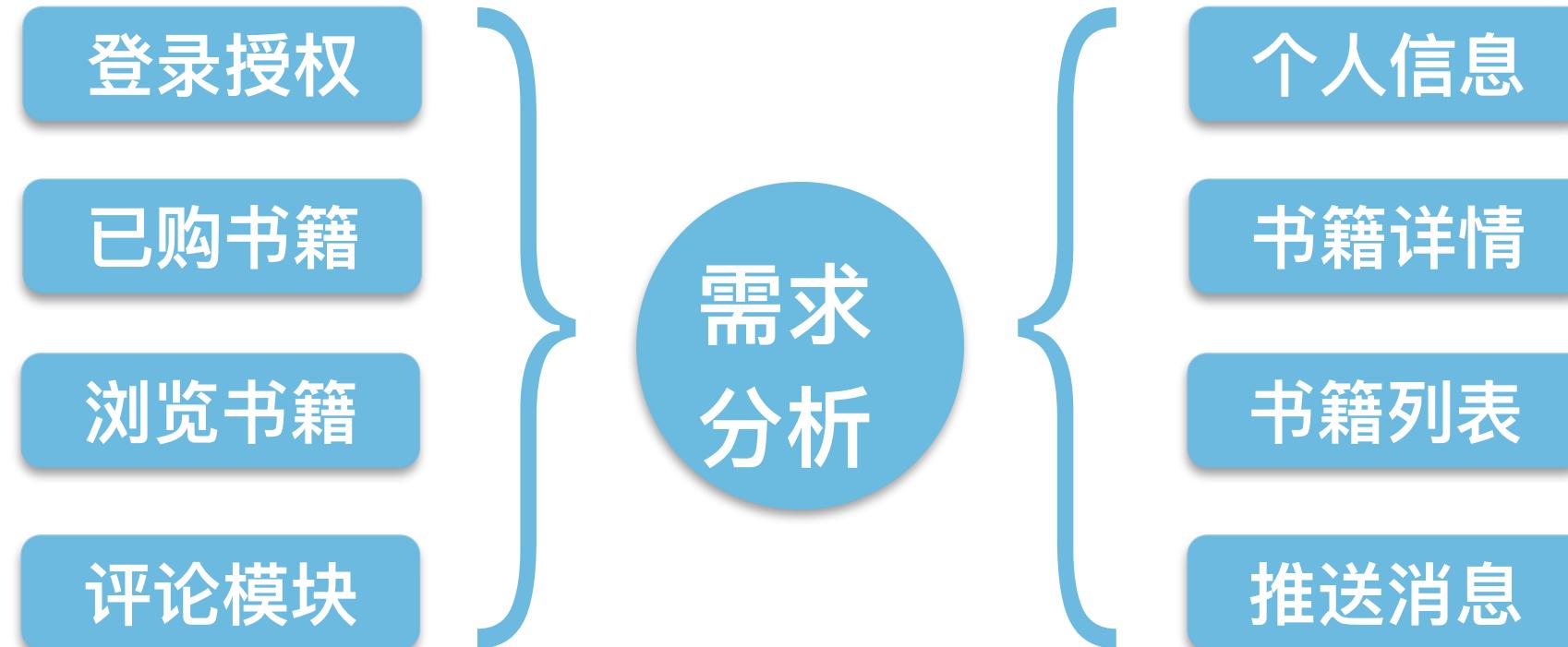


# 微信小程序核心功能——打开APP

- 主动触发
- 同一微信开放平台账号
- 只支持分享至好友会话

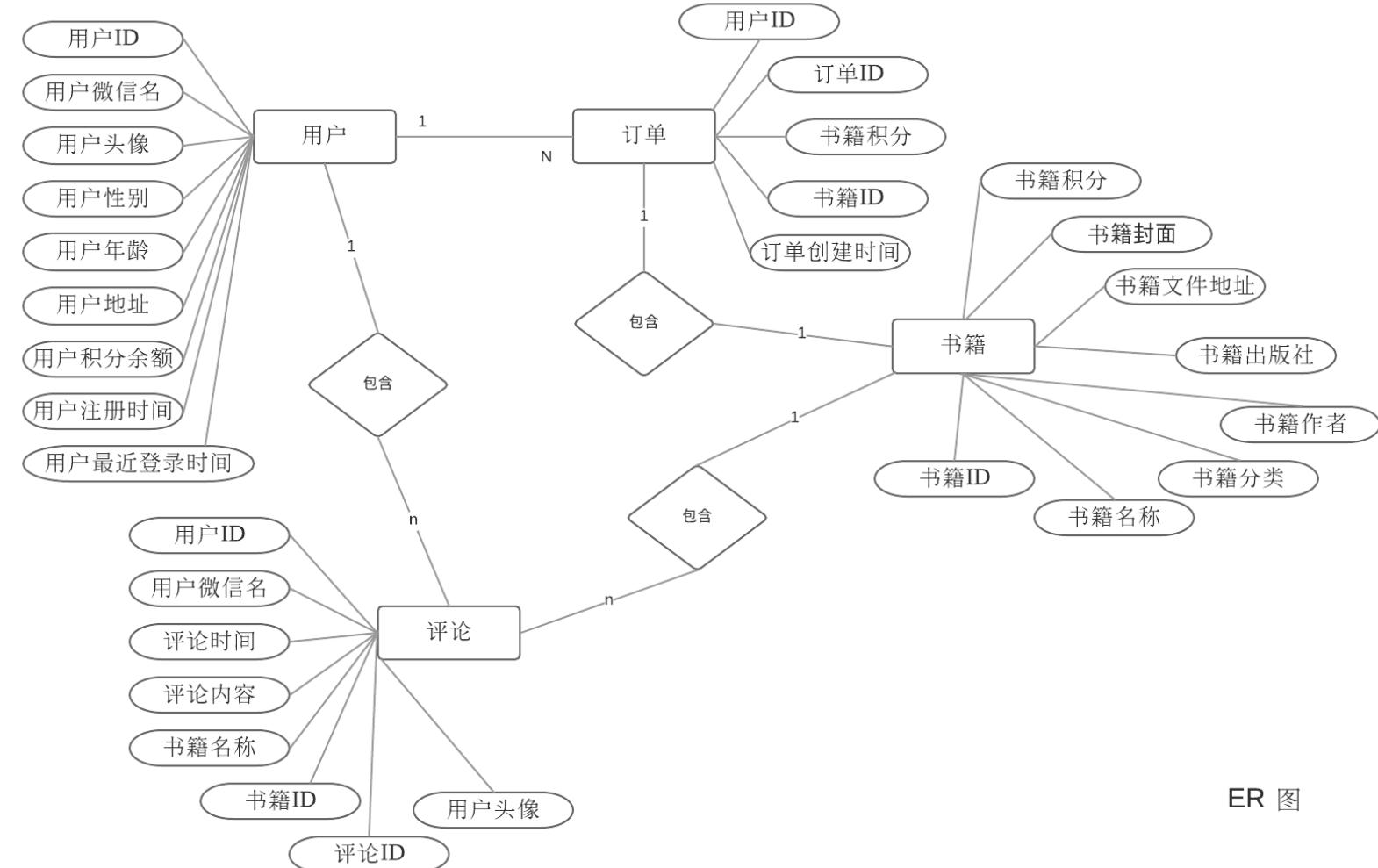


# 微信小程序实战项目——小书架



# 微信小程序实战项目——小书架

E-R 图



ER 图

# 微信小程序实战项目——小书架



users

books

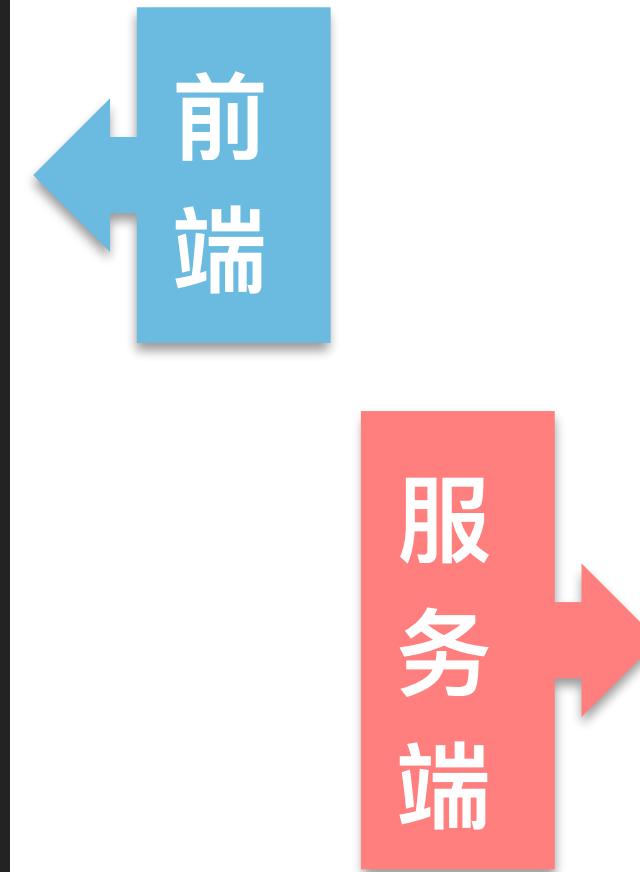
comments

orders

access

# 微信小程序实战项目——小书架

```
› config
› images
└ pages
    › books
    › comment
    › detail
    › logs
    › my
    › myBooks
    › utils
JS app.js
{} app.json
# app.wxss
{} project.config.json
```



```
› bin
› conf
› controllers
› dao
› middleware
› node_modules
› routes
└ util
diamond .gitignore
JS app-token.js
JS app.js
npm npm-debug.log
{} package.json
```

# 微信小程序组件化开发框架——wepy



## 开发风格

接近于 Vue.js，支持组件 Props 传值，自定义事件、组件分布式复用Mixin、计算属性函数 computed、模板内容分发slot等等



## 组件化

组件化开发，完美解决组件隔离，组件嵌套，组件通信等问题



## NPM

支持使用第三方 npm 资源，自动处理 npm 资源之间的依赖关系，完美兼容所有无平台依赖的 npm 资源包



## Promise

通过 polyfill 让小程序完美支持 Promise，解决回调烦恼



## ES2015

可使用 Generator Function / Class / Async Function 等特性，大大提升开发效率



## 优化

对小程序本身的优化，如请求列对处理，优雅的事件处理，生命周期的补充，性能的优化等等



## 编译器

支持样式编译器：Less/Sass/Styus，模板编译器：wx-mi/Pug，代码编译器：Babel/TypeScript。



## 插件

支持多种插件处理，如文件压缩，图片压缩，内容替换等，扩展简单，使用方便

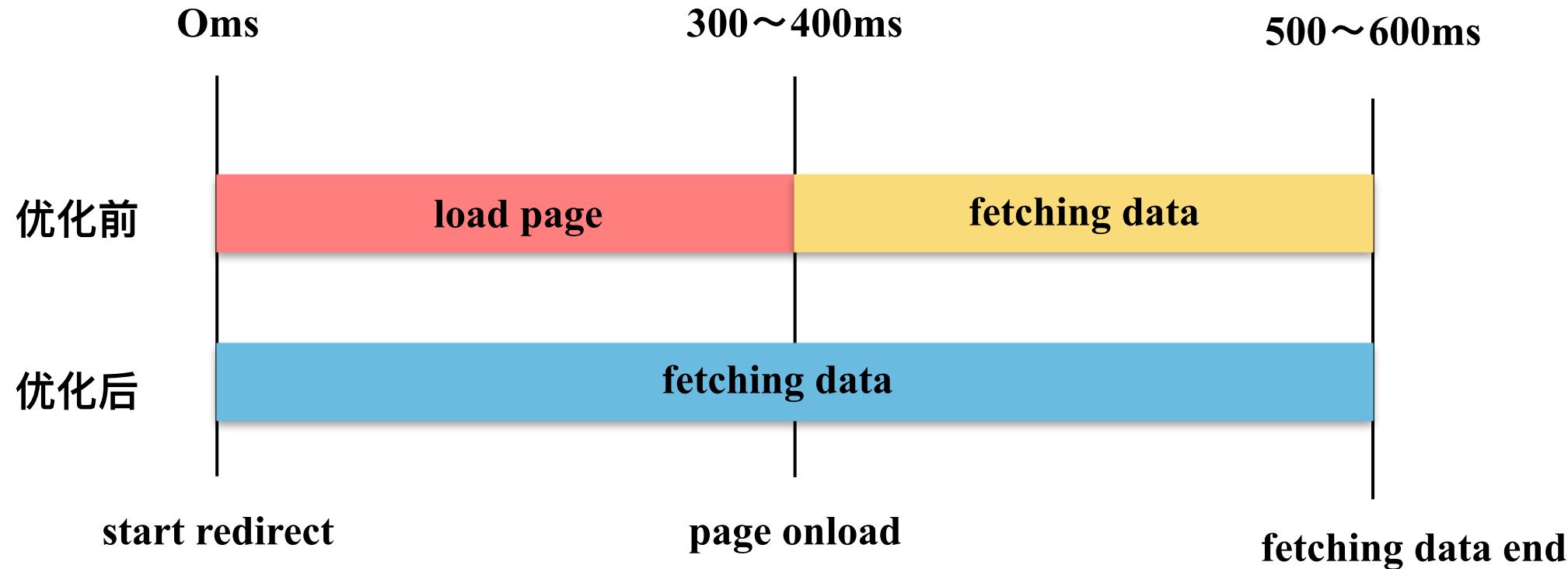


## 框架大小

压缩后 24.3KB 即可拥有所有框架功能，额外增加 8.9 KB 后即可使用 Promise 和 Async Function

# 微信小程序组件化开发框架——优化

## 提高页面加载速度



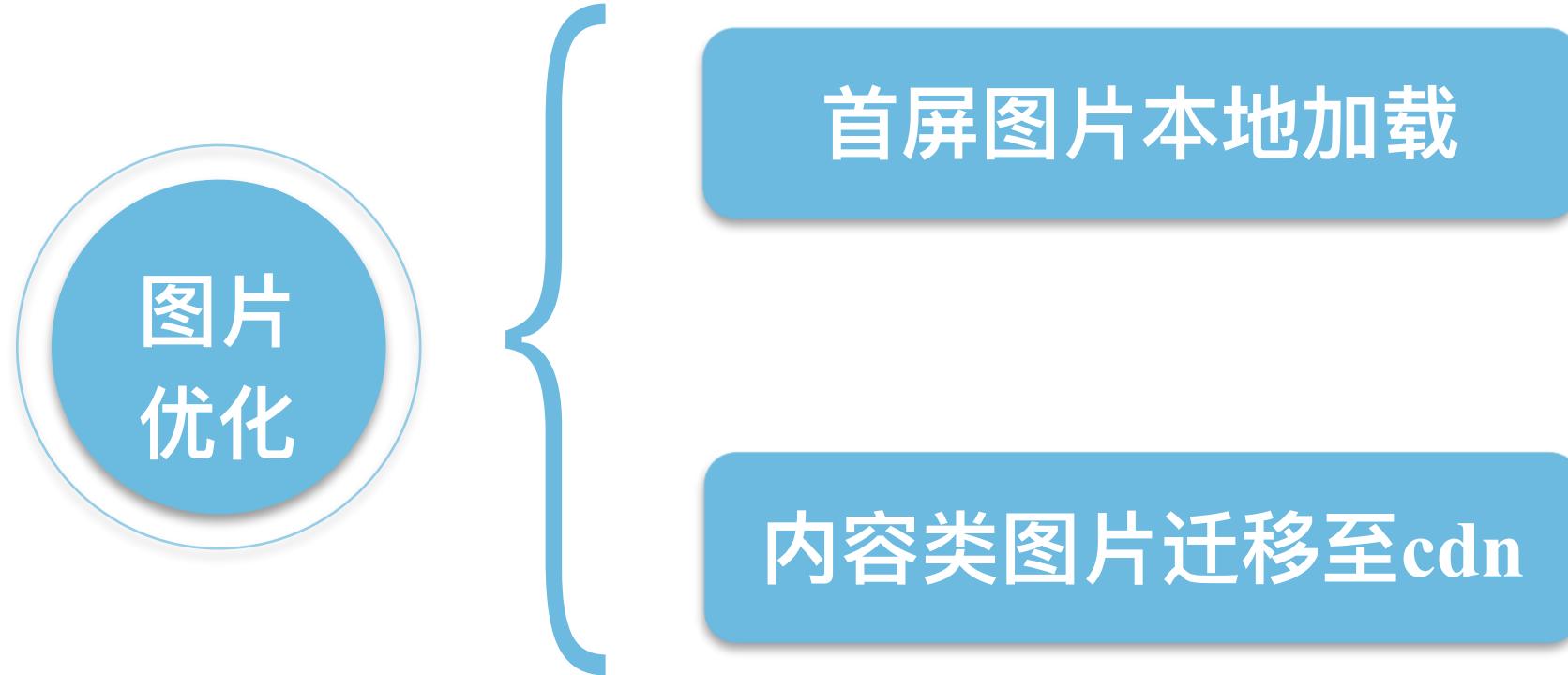
# 微信小程序组件化开发框架——优化

## Promise异步解决方案

```
const fPromise = fn => obj => {
  return new Promise((resolve, reject) => {
    obj.complete = obj.success = (res) => {
      // console.log(res,'>')
      resolve(res)
    }
    obj.fail = (err) => {
      reject(err)
    }
    fn(obj);
  })
}
```

```
fPromise(wx.getUserInfo)({})
  .catch(e => {
    console.log(e);
  })
  .then(res => {
    // get userinfo
    return fPromise(wx.downloadFile)({
      url: res.userInfo.avatarUrl
    });
  })
  .then(res => {
    // get avatar tempfilepath
    return fPromise(wx.saveFile)({
      tempFilePath: res.tempFilePath
    });
  })
  .then(res => {
    console.log(res.savedFilePath)
  })
})
```

# 微信小程序组件化开发框架——优化



# 微信小程序组件化开发框架——分包加载



首次启动，先下载  
小程序主包

进入某个分包页面时，  
下载对应分包