概要设计-示例-微信小程序登录模块

> 💡 本文仅作为 **概要设计示例** 使用，并非目前公司小程序登录模块实际设计。

一、背景

小程序可以通过微信官方提供的登录能力方便地获取微信提供的用户身份标识，快速建立小程序内的用户体系。

用户身份标识包括以下两种：

- `OpenID`：用户在小程序、公众号里的唯一标识，**仅限于同一应用**
- `UnionID`：同一用户，对同一个微信开放平台下的**不同应用**，UnionID是相同的

此外，一个完整的应用通常会包含用户的基本信息，可以通过小程序提供的「用户信息接口」获取除用户标识外的其他信息（如：昵称、头像）。（需要特别注意的是，**为优化用户登录体验，微信官方对该接口进行了调整，详见：用户信息接口调整说明**）

虽然在「微信生态」中可以很方便地建立成熟的用户体系，但同时也要考虑 微信小程序设计指南 中提及的用户体验。总之，好的登录流程应该做到：**足够友好**。

场景：

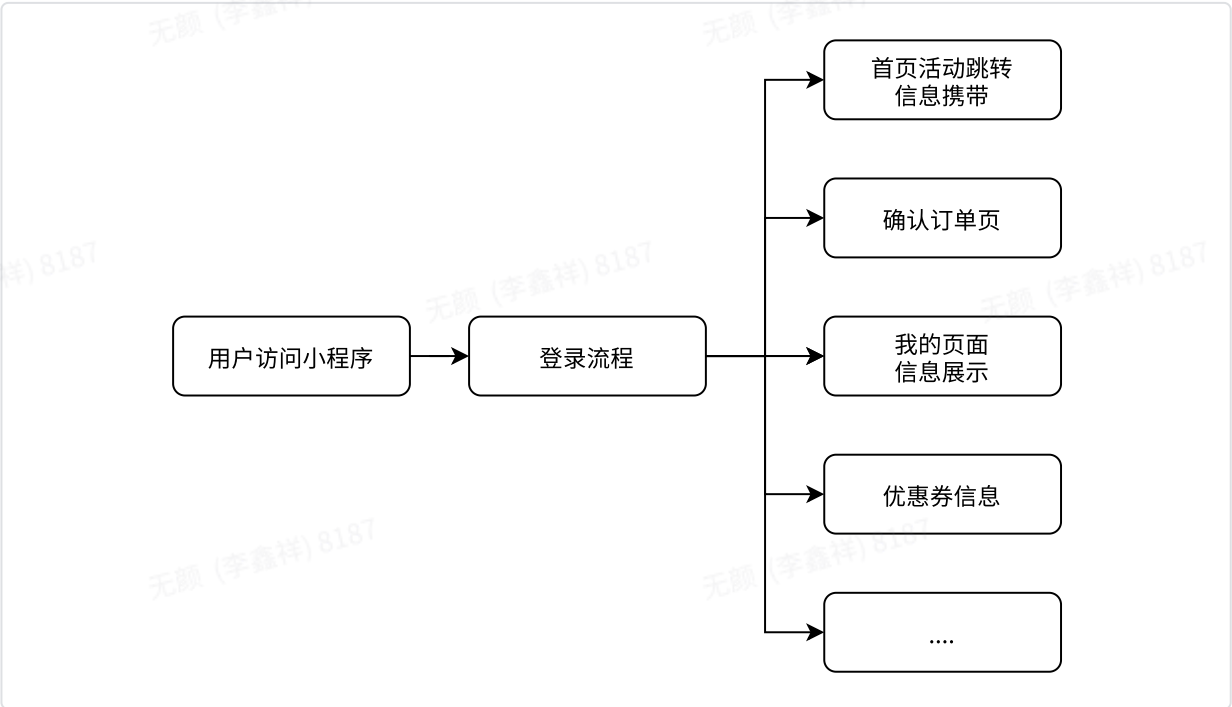
- 是否有多个应用 （否）
- 是否需要用户基本信息（需要，但仅用于展示）

目的：

- 建立营销体系（如：会员、营销、客服系统）
- 建立监控体系 （如：埋点）
- 提升用户体验 （不限于登录）

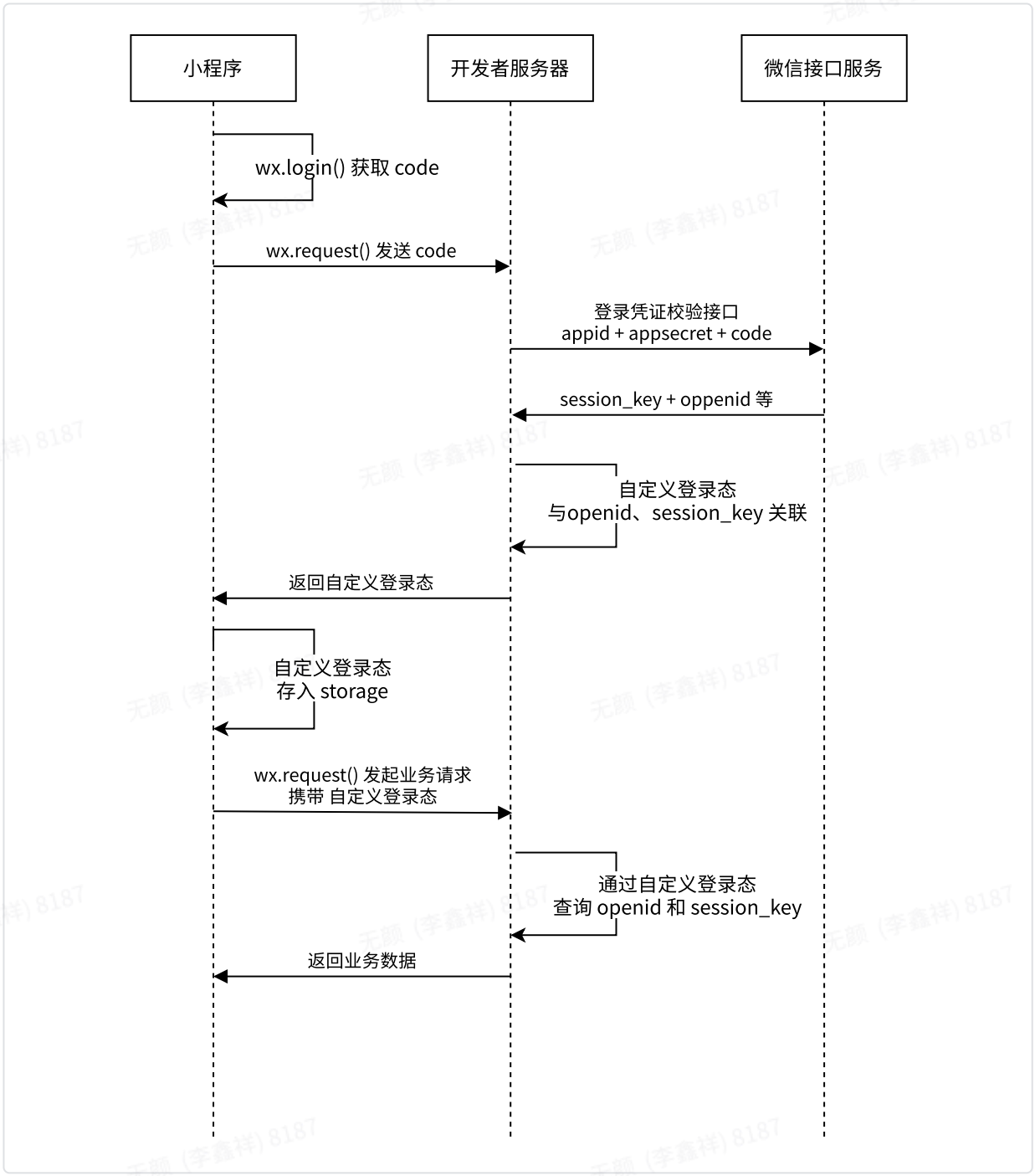
二、关键业务流程

结合以上背景的描述，初步拟定本文采用一种「 静默登录 」的方式来完成登录流程（所有流程基于官方登录)。



2.1 官方登录

首先，来看一下微信官方提供的登录流程时序：



总结步骤为：

1. 调用 `wx.login()` 获取 **临时登录凭证 code**，并回传到开发者服务器。
2. 调用 `auth.code2Session` 接口，换取 **用户唯一标识 OpenID** 和 **会话密钥 session_key**。

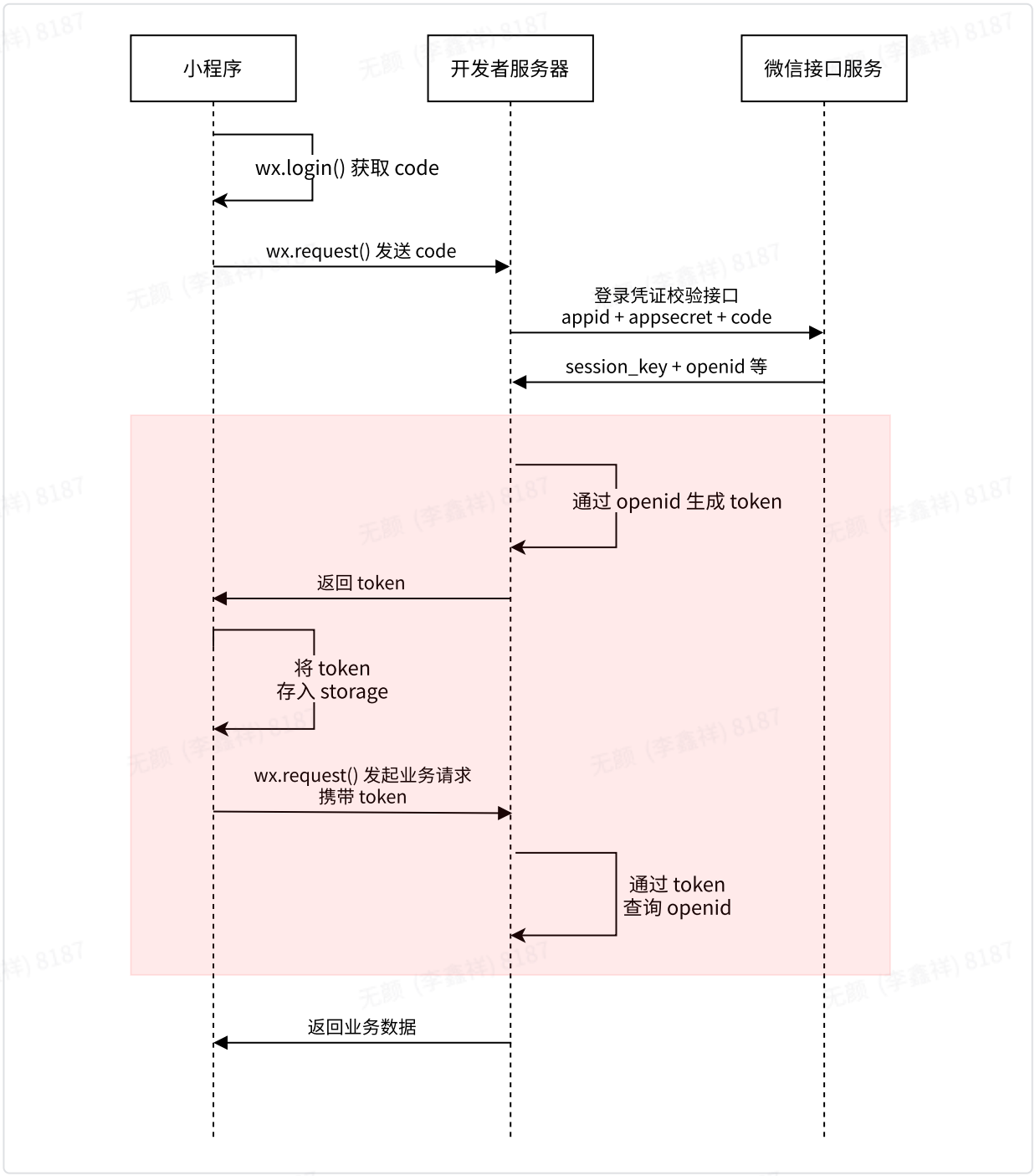
之后开发者服务器可以根据用户标识来生成**自定义登录态**，用于后续业务逻辑中前后端交互时识别用户身份。更多细节，详见 小程序登录官方文档

2.2 自定义登录态

当开发者在实现自定义登录态时，可以考虑以 **session_key** 有效期作为自身登录态有效期，也可以实现自定义的时效性策略。

考虑到 **session_key** 不能做到 100% 同步用户登录状态，以及减少对微信生态的依赖，我们选用自定义的方式来维护登录态，即：通过用户唯一标识 **OpenID** 生成 **Token** （用于 JWT 接口鉴权），登录过期机制与服务端接口授权过期一致。

自定义登录态 (token) 的登录流程：



需进一步确认：

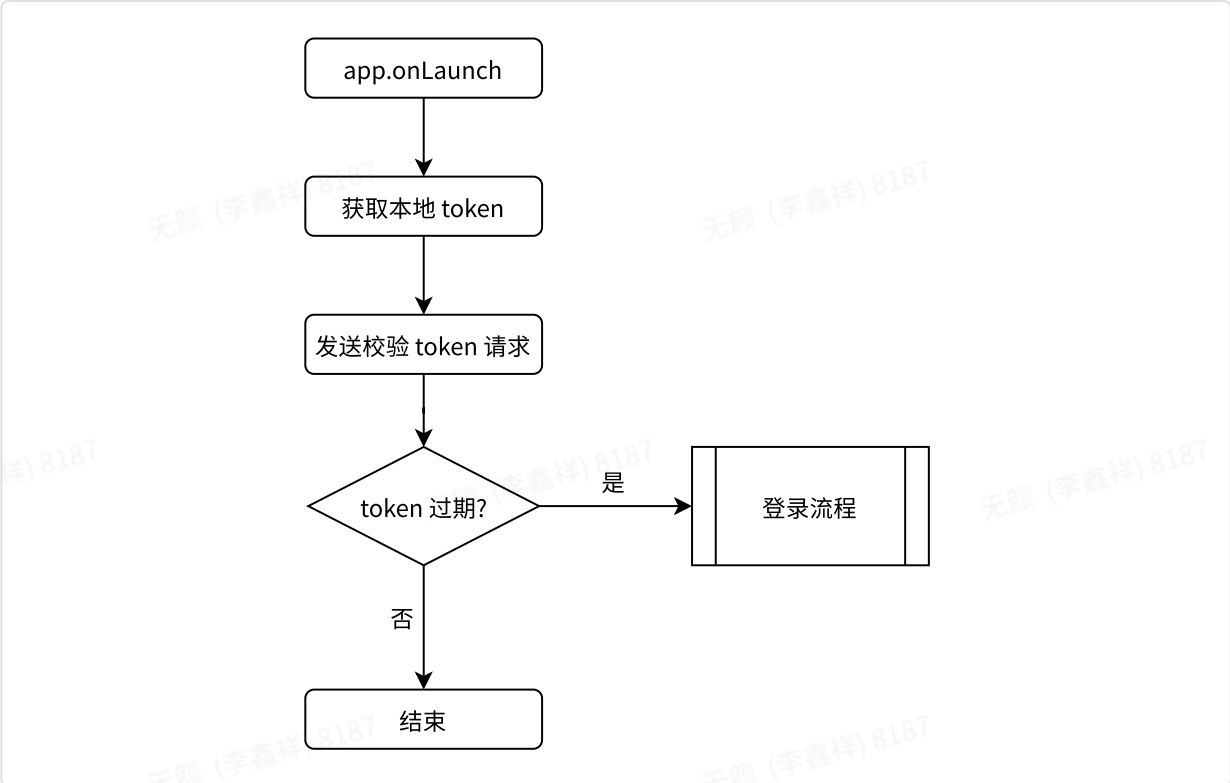
- 仅通过 openid 能否生成 token？
- 用户信息传给服务端的时机？

注意：微信小程序的执行环境没有 cookie，故建议采用 JWT 形式

2.3 接口鉴权

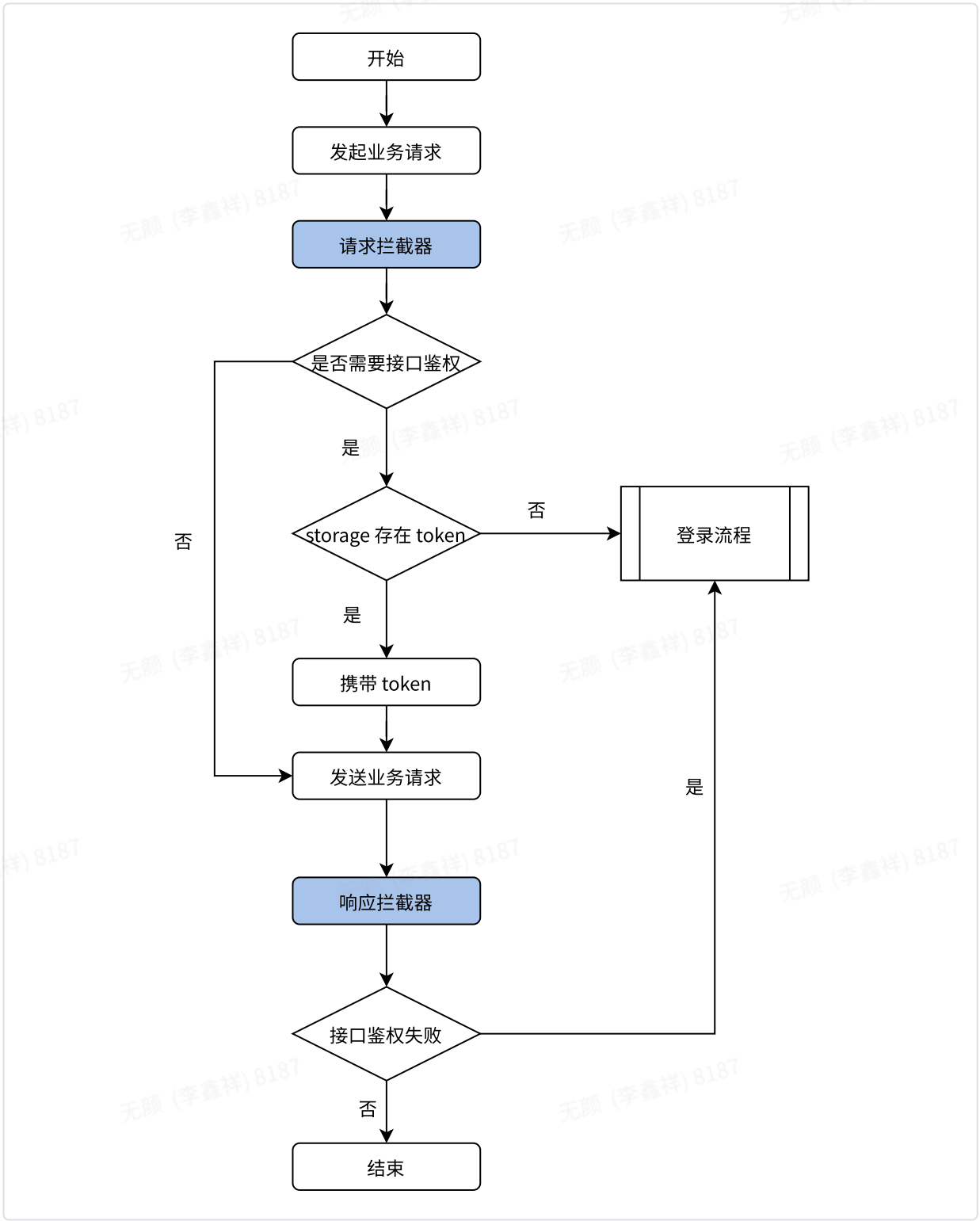
接口鉴权的时机有两种：小程序启动时、接口请求发起时。

- **小程序启动时鉴权：**



需要注意的是：小程序的生命周期钩子函数不支持异步阻塞，可能出现页面加载完先于静默登录完成的情况，将导致需要鉴权的接口请求失败。

- **接口请求发起时鉴权：**

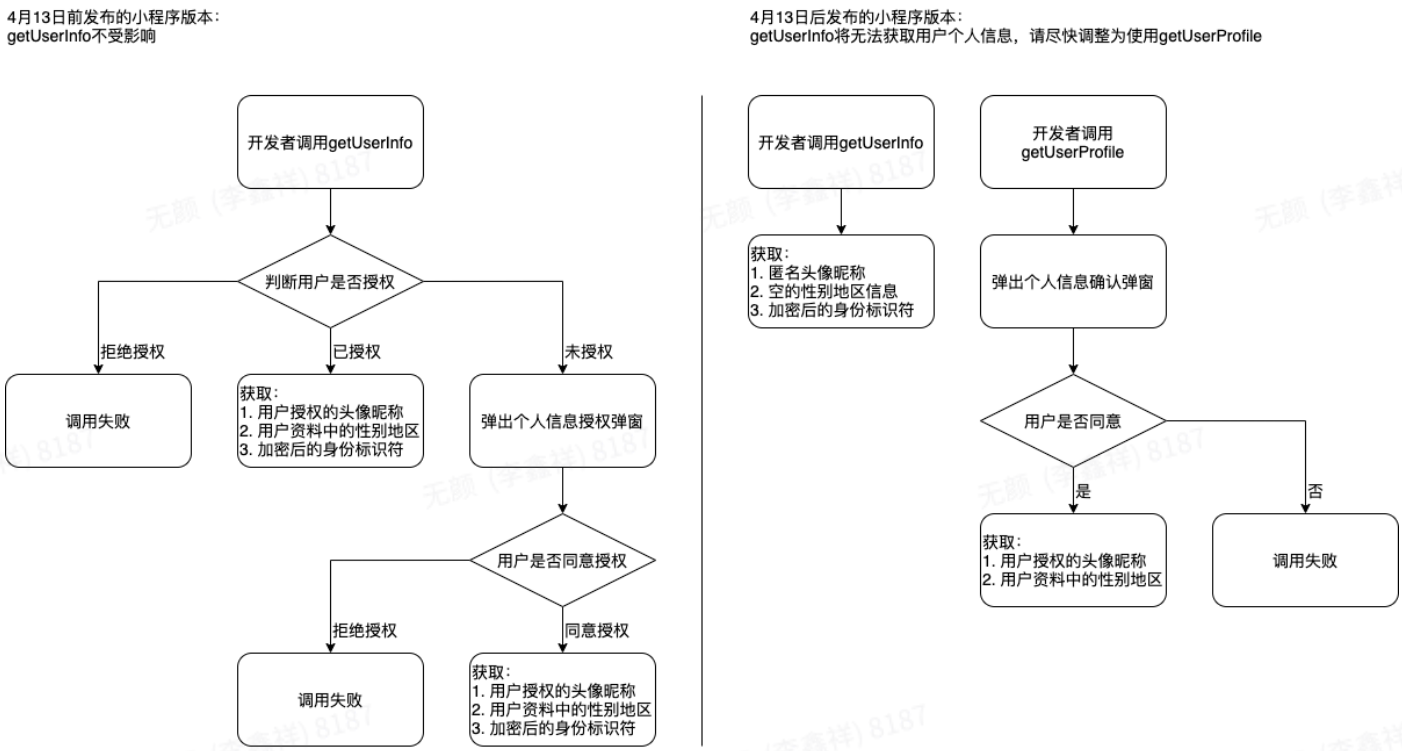


需进一步与服务端确认**接口鉴权策略**，同时确定登录流程。对于接口鉴权的**并发处理**可深入讨论后完善对应流程。

2.4 用户信息获取

为提升用户体验微信官方对用户接口进行了调整，详见 **用户信息接口调整说明**

调整后流程如下：



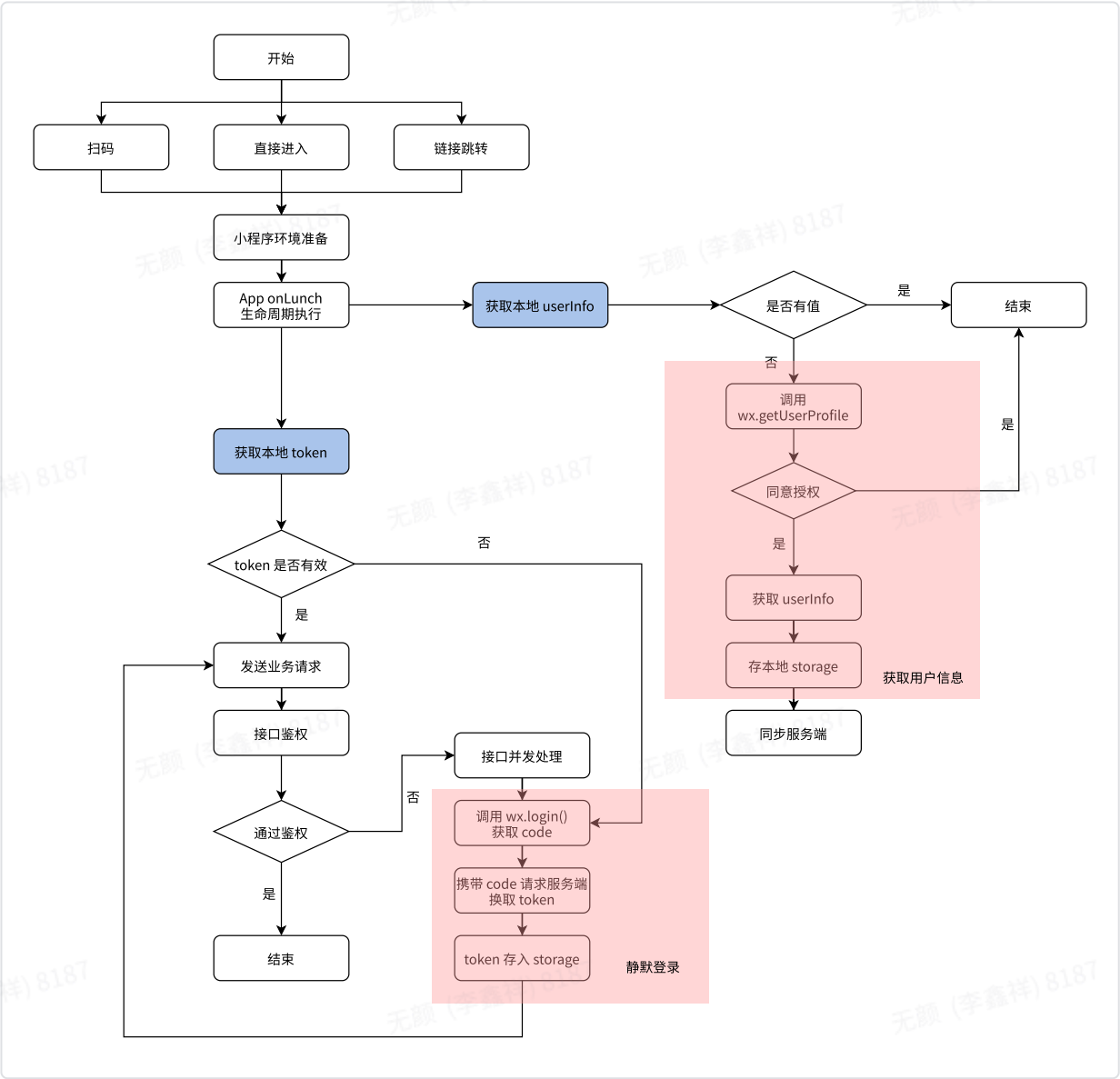
总结如下：

1. 调整后，开发者如需获取用户身份标识符只需要调用 `wx.login` 接口即可。
2. 开发者若需要在界面中展示用户的头像昵称信息，可以通过 `<open-data>` 组件进行渲染，该组件**无需用户确认**，可以在界面中直接展示。
3. 开发者需要获取用户的头像昵称信息，可调用 `wx.getUserProfile` **接口**，每次通过该接口均需用户确认，请开发者妥善处理调用接口的时机，**避免过度弹出弹窗骚扰用户**。

2.5 总结

对于登录主要分两个流程：

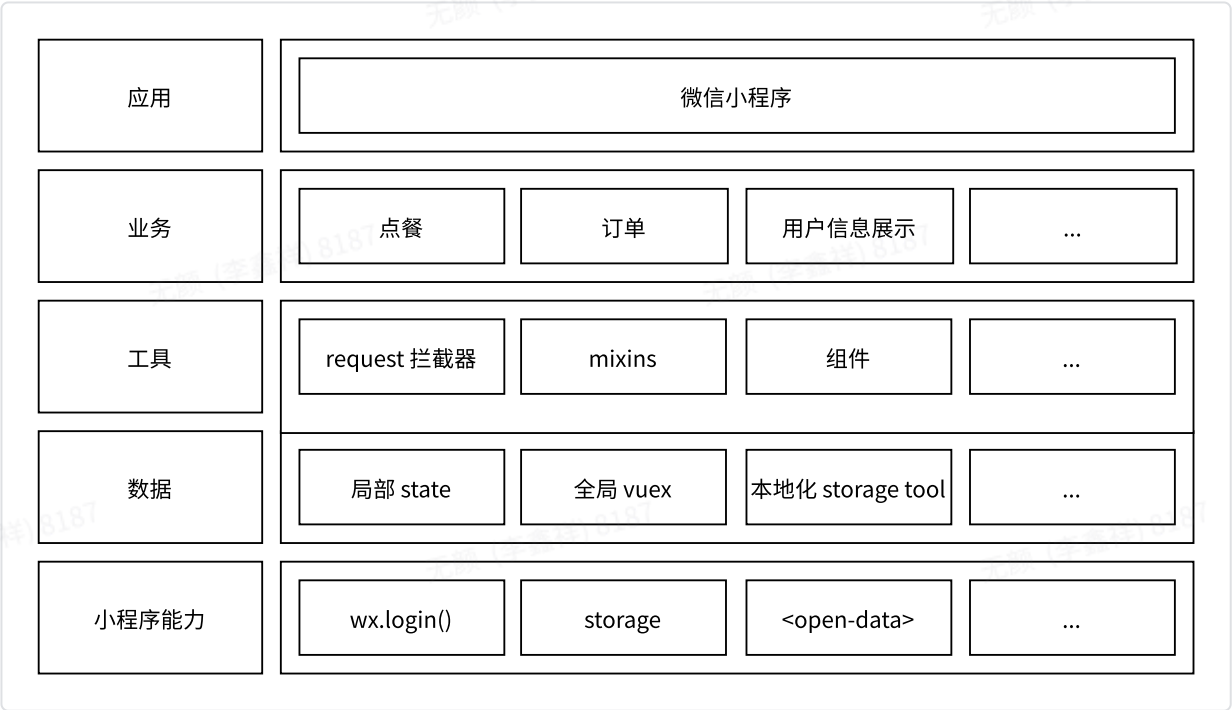
1. 获取接口鉴权 token
2. 获取用户信息，关联用户 openid



注意：对于常规的用户信息展示，可以通过 `<open-data>` 组件完成。所以，理论上，第一次获取用户信息同步服务端后，不再需要同步，这里涉及用户数据更新策略。

三、关键技术描述

3.1 整体架构



整体架构依赖于微信小程序提供的平台能力、API 服务、组件等，数据维护基于 vue 的 state、vuex 的 全局 store，并提供 request 拦截器接口鉴权、授权通用组件、数据交互通用 mixins 等，服务于点餐、订单、用户信息等模块。

3.2 技术点

小程序基础能力：

- `wx.login`
- `wx.getUserProfile`
- `wx.checkSession`
- storage

数据维护：

- 本地 storage
- 全局数据 Vuex

辅助工具：

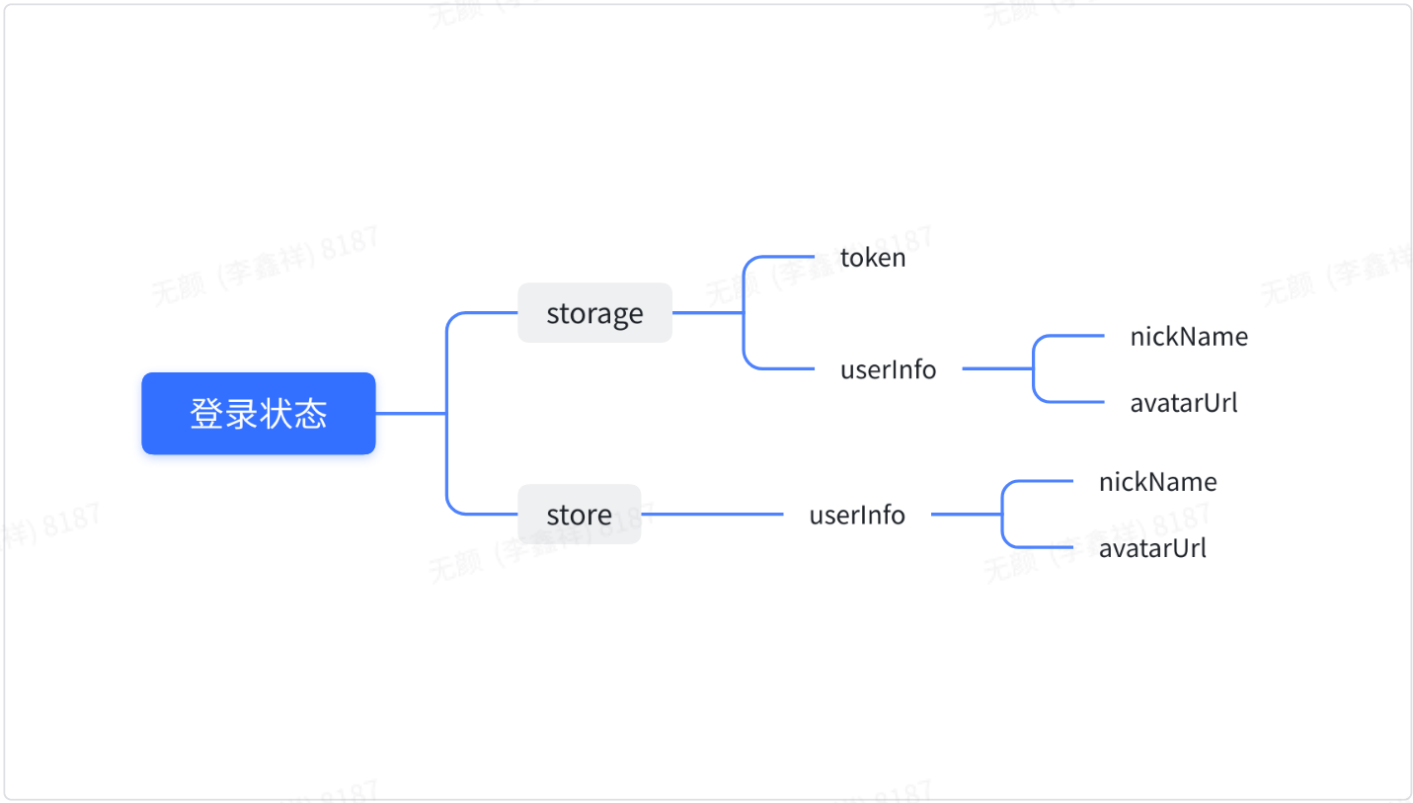
- Request 拦截器，参考：Axios Interceptors
- Mixins

· 自定义组件

四、模块拆分

五、状态设计

对于登录来说，状态并不多，设计如下：



注意微信官方接口调整后的数据变化：

属性	说明	4月13日前发布新版本的小程序	4月13日后发布新版本的小程序

userInfo	用户信息对象	{
		"nickName": "用户授权的昵称",
		"avatarUrl": “用户授权的头像”,
		"gender": 用户微信资料中的性别,
		"country": “用户微信资料中的国家”,
		"province": “用户微信资料中的省份",
		"city": “用户微信资料中的城市”,
		“language”: “用户微信资料使用的语言”
		}
			{
			"nickName": "微信用户",
			"avatarUrl": “灰色头像”,
			"gender": 0,
			"country": “”,
			"province": “",
			"city": “",
			“language”: “”
			}

rawData	不包括敏感信息的原始数据字符串	用户授权的userInfo的字符串格式	匿名userInfo的字符串格式
signature	使用 sha1(rawData + sessionkey) 得到字符串，用于校验用户信息	生成规则不变	

encryptedData	包括敏感数据在内的完整用户信息的加密数据	解密后得到的数据：
		{
		"openId": "OPENID",
		"nickName": "用户授权的昵称",
		"gender": 用户微信资料中的性别,
		"city": "用户微信资料中的城市",
		"province": "用户微信资料中的省份",
		"country": “用户微信资料中的国家”,
		"avatarUrl": “用户授权的头像”,
		"unionId": "UNIONID",
		"watermark":
		{
		"appid":"APPID",
		"timestamp":TIMESTAMP
		}
		}
			解密后得到的数据：
			{
			"openId": "OPENID",
			"nickName": “微信用户”,
			"gender": 0,
			"city": "",
			"province": "",
			"country": "",
			"avatarUrl": “灰色头像”,
			"unionId": "UNIONID",
			"watermark":
			{
			"appid":"APPID",
			"timestamp":TIMESTAMP
			}
			}

iv	加密算法的初始向量	不变	
cloudID	敏感数据对应的云 ID	云调用后获取的数据见encryptedData	

六、接口设计

> 💡 如果涉及自定义类，可以在这里提及接口设计