# 1、Android 原生开发

原生开发依然是开发者采用最广泛的开发方式，优点十分显著。相比其他开发方式而言，原生开发可以访问设备中的所有功能，运行速度更快，性能更高，而且可以启用优秀的离线处理和存储能力等等，提供最佳的用户体验，最优质的用户界面，最华丽的交互。原生开发人员众多，开发环境成熟，有许多的开源库提供开发人员调用，可是方便实现各种设计效果。

原生开发的缺点在逐渐的开发、运营过程中显现出来。开发成本高，不同平台需要定制不同的app，也就是android定制apk，ios定制app，开发人员需要多平台多语言，人力成本、时间成本较多，通用性差；上线时间不稳定，需要审核，特别是苹果审核机制，审核时间长短不一，对内容还有控制，国内Android APP市场（百度手机助手，应用宝，360市场等等）也有类似的问题；版本控制能力差，版本发布到达率无法控制，多个版本更新发布，修复bug，无法保证及时送达到用户手中；获得新版本需要重新下载安装，虽然目前有增量升级方式逐渐改变，但是随之而来的其他问题如增量升级多版本控制也是个很头疼的问题；

## 总结 原生开发虽然有各种缺点，但是在目前所有的开发技术中原生是最成熟，有效，也是开发人数最多，开源库最广泛的。对APP要求各方面性能、响应要求高，人员充足，完整开发、测试流程，适合原生APP开发。

# 2、HTML5+MUI +HBuilder

## MUI介绍

MUI样式地址：<http://dev.dcloud.net.cn/mui/>

MUI（Mobile User Interface）是一套基于Html5的，遵循Html5+规范的。中国团队（是否有老外我不知道哦）开发的，开源的（遵循MIT 条款），用于手机端界面开发的一套框架。Github地址

https://github.com/dcloudio/mui

## 存在问题：

主要体现在：性能跟不上、mui官方各种bug出来没人修复，社区越来越不活跃、有很多三方的插件官方集成难度太大。

H5加载大图片的时候性能会下降，大量用户访问同一个H5应用时性能会下降，响应速度比不上原生app，上网速度也不及原生app，H5不能自动处理动画上反复交互（网页游戏），需要借助css3、javascript。H5本质上是网页，无论是离线的还是在线的，它本质上是通过浏览器呈现到用户面前的网页，在手机上模拟得像app，网页的缺陷它无可避免。

1. 性能问题，这才是H5最大的问题，原生开发对性能的支持远超H5，在用户体验上，H5的app绝对是占据下风的，app反应速度、流畅度等；
2. 功能问题，对某些硬件摄像头、陀螺仪、麦克风等硬件支持较差，频繁调用这些硬件，H5不容易扩展，调用速度也不如人意。

总结

1. 纯H5 app适合小游戏、媒体、视频、营销产品、介绍等功能，大部分大型app属于原生、H5混合的hybird app。

# 3、Weex

## 介绍：

文档不够完善、组件也不够丰富、因为涉及到很多原生层面代码所以需要有一定的android和ios原生功底，不然到时候遇到bug都不知如何下手了、还有就是社区不活跃。

Weex 致力于使开发者能基于通用跨平台的 Web 开发语言和开发经验，来构建 Android、iOS 和 Web 应用。简单来说，在集成了 WeexSDK 之后，你可以使用 JavaScript 语言和前端开发经验来开发移动应用。

Weex 渲染引擎与 DSL 语法层是分开的，Weex 并不强依赖任何特定的前端框架。目前 [Vue.js](https://vuejs.org/) 和 [Rax](https://alibaba.github.io/rax/) 这两个前端框架被广泛应用于 Weex 页面开发，同时 Weex 也对这两个前端框架提供了最完善的支持。Weex 的另一个主要目标是跟进流行的 Web 开发技术并将其和原生开发的技术结合，实现开发效率和运行性能的高度统一。在开发阶段，一个 Weex 页面就像开发普通网页一样；在运行时，Weex 页面又充分利用了各种操作系统的原生组件和能力。

## 存在问题：

文档不够完善、组件也不够丰富、因为涉及到很多原生层面代码所以需要有一定的android和ios原生功底，不然到时候遇到bug都不知如何下手了、还有就是社区不活跃。

# 4、React-Native：

## 介绍

react-native的优点和H5类似，跨平台、低成本开发、开发速度快、动态发布、一套类似代码维护三个平台

react-native不是万能药，只是一种高效的解决方案，不要期望它解决所有的问题，要不要使用需要场景衡量；客户端转开发react-native感觉比较简单，比较JS大部分人都有基础，前端人员开发react-native理解原生部分的代码应该十分困难；相比原生海量的第三方控件和第三方包，react-native大部分道路还得靠自己摸索；不同端的代码风格和结构大体类似，但具体标签可能会不一样；目前得到经验是IOS版本比较稳定，android版本还不太成熟，android 版本2015年下半年发布，一直在0.\*版本上更新，android1.0版本还没有发布；把把facebook的第三方包网络连接包okhttp和图片下载解析包fresco等一起封装进结果，造成包增加7、8M，但据了解这是可以修改的；只支持IOS7以上和android4.1以上机型，这应该不成问题，比较其他属于少数；听说qzone使用了react-native，但是crash率前10，react-native占8个，前5全是react-native，但是我一朋友项目使用过react-native，虽然有坑，但是不会有如此多crash;

## 优点：

组件丰富（官方插件组件很丰富、三方插件只有你想不到的没有做不到的、基本形成三方插件和官方插件互补的形式）、最重要的是原生功底没有硬性要求，一份代码能够兼容多端，这样就能够大大提升开发速度了、大部分与原生代码交互直接使用三方插件配置即可。

**5、Cordova**

Cordova是一个开源的移动开发框架。允许用户使用标准的web技术HTML5，CSS和JavaScript做开发平台，应用在每个平台的具体执行被封装了起来，并依靠符合标准的API绑定去访问每个设备的功能，比如说；传感器、数据、网络状态等。（摘自Cordova官方中文文档）。

说白了，Cordova其实就是内嵌一个移动端的web页面，依托于原生平台上实现页面展示，业务逻辑处理等功能。Cordova生成的工程是原生App里内嵌一个UIWebView

## 缺点

1、浏览体验短期内还无法超越原生应用；

2、不支持离线模式（HTML5将会解决这个问题）；

3、消息推送不够及时；

4、调用本地文件系统的能力弱。