一、WebView流

WebView本身可以作为显示web页面的容器，而它本身就是跨平台的，因此利用WebView来进行跨平台的技术方案就称为WebView流。

而WebView流本身也经历了三个阶段

1、纯H5

就是在app中嵌入一个webview，而native与h5是几乎没有任何交互的，因此native无法使用h5特性，同样h5也无法使用native特性，这些限制了webview使用场景

2、Hybrid（中文含义：混合的）

为了打破Native和H5的割裂情况出现了JSBridge，它是Native与JS代码的通信桥梁。这样H5和native可以使用彼此的特性。这种H5和native彼此混合交互的跨平台方案称为

Hybrid。

优点：开发成本低、简单、跨平台

缺点：内存占用多、网页加载缓慢、渲染慢、JS执行慢等性能问题

应用场景：主要使用native开发，少数界面使用H5

3、基于Hybrid的性能优化问题

针对具体存在的问题进行优化：

1、网页加载速度慢——采用离线包的方案（参考链接 https://blog.csdn.net/LuckyWinty/article/details/103790120）

大体原理是在打包app时讲静态资源打包到安装包中，请求时查找本地静态资源是否存在，存在则直接返回，否则请求

2、网页渲染慢——采用优化dom树

但是由于webview的渲染能力比native，同时js时解释执行语言，运行效率比native差（见尾页解释）

二、ReactNative流

RN流抛弃了webview，使用js开发，典型方案有RN和Weex

1、跨平台的布局引擎

内置了跨平台的布局引擎可以将H5的布局转化为native布局

2、使用native原生组件渲染

ReactView组件使用原生组件渲染

3、JS引擎

内置JS引擎可以在不同平台上运行js代码

整体来说，RN流采用的是：js+js引擎+native技术方案，主要是利用了js来进行跨平台开发

三、编译流/虚拟机流

主要代表方案是Xamarin，主要是应用在国外

使用C#开发，在ios上会被直接编译为Native ARM code,而在Android平台则是运行在Mono虚拟机上，而Mono虚拟机泽运行在Android平台上。这种通过第三种语言开发编译到目的平台运行称为编译流或者虚拟机流。

也就是说，Android平台是使用虚拟机运行，ios由于苹果限制，无法使用虚拟机，因此会编译为ARM code，因此称为编译流/虚拟机流

而UI渲染也是映射为Native原生组件来渲染

四、游戏引擎流

游戏引擎例如Unity等，而游戏引擎的逻辑和渲染都是由游戏引擎自己负责，而开发普通app则是大材小用。

主要原因如下：

1、游戏引擎的安装包体积很大

2、游戏引擎和普通app刷新机制不同，游戏引擎是实时刷新，普通app是按需刷新，而且实时刷新耗电量较高

3、游戏引擎虽然可以做出复杂的游戏界面，但是Android和iOS,原生界面需要自己重新开发，

4、游戏引擎适合开发一个新的app，不适合和原生app做混合开发

五、理想的跨平台开发方案

理想的跨平台方案需要解决的技术难题：

- 一种是逻辑代码，逻辑代码需要语言来实现
- 一种是UI，要么自己渲染要么使用原生渲染

Webview流使用js开发，js逻辑运行在webview容器，UI使用webview自己的渲染。
RN流使用js开发，js逻辑运行在js引擎，使用原生渲染。
Xamarin使用c#开发，UI使用原生渲染。
游戏引擎，逻辑代码和UI都是运行在游戏引擎容器里。

六、如何设计一套自己的跨平台方案

原则：

1、只是用一种语言开发，而不是原生+js等

2、UI最好自己来最渲染，有自己一套渲染引擎，并且渲染性能和原生一致。

3、可以和对应平台有良好的交互性，即可以使用原生平台的特性，例如调用相机、申请权限

七、Flutter

1、使用Dart语言开发，支持两种编译方式（JIT和AOT），不同场景下使用不同的编译方式，对应提高开发效率和运行体验

debug模式下，使用JIT动态编译，将代码运行在虚拟机上，使我们编写的代码实时更新，实现hotReload特性，提高开发效率

release模式下，采用AOT静态编译模式，将代码直接编译为平台对应的native代码，保证运行速度和渲染速度，提高用户体验

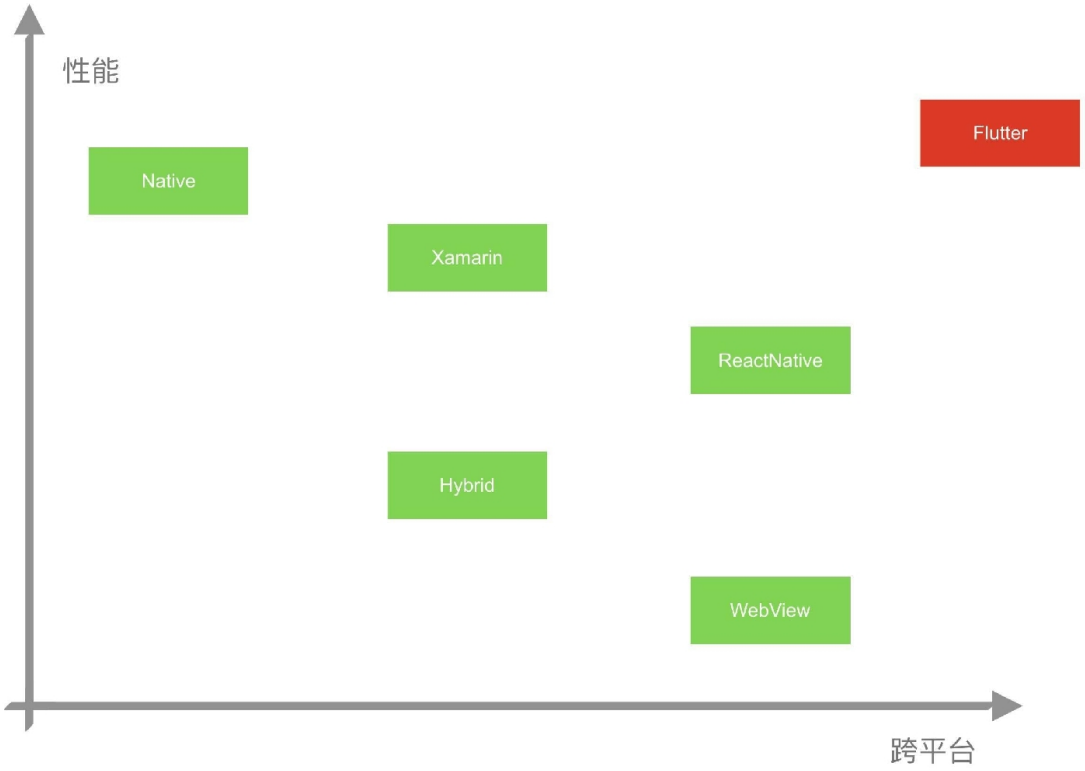
2、UI渲染方面，不依赖于平台，自带skia渲染引擎

3、与平台交互上，提供platform channel通道，便于与native交互

4、google团队维护，文档完善

八、Flutter与其他跨平台方案比较

性能上接近原生，跨平台实现上也表现优异



这里的对比不是很完善，后期应该有详细的对比图

九、Flutter展望

flutter不仅仅是一套UI框架，也是google新一代操作系统Fuchsia /ˈfjuːʃə/的UI框架

Fuchsia基于Zircon的微内核，设计的目标是可运行在众多设备上。

问题：

1、解释执行语言比编译执行语言效率低的原因

（1）、解释执行语言

优点：

解释执行不依赖与平台，编译器会根据不同的平台进行解析

开发速度快

缺点：

解析需要时间，不会生成目标程序而是一句句执行，浪费计算机资源

（2）、编译执行语言

优点：会先编译，执行效率高，占用资源少

缺点：兼容性差，与平台相关，在不同平台上可能存在运行问题