React Native的开发实践

黄治豪
网易游戏-前端工程师

1. 背景
2. 工程实践
3. 总结与思考

背景

1. 旧项目使用全站 H5

2. 性能瓶颈

3. 需要成本不高的解决方案（相对于全原生开发）

背景 - 预研

	Native	Hybrid	Weex	React Native
原生体验	优秀	良好	接近优秀	接近优秀
更新复杂度	高	低	较低	较低
编程语言	Java，OC/Swift	JS	JS + Vue	JS + React
社区资源	丰富	一般	中等	丰富
上手难度	高	低	中等	中等
开发效率	低	高	较高	较高

背景 - 预研

Weex

阿里巴巴，2016年6月开源

Write Once Run Everywhere

支持Android，IOS，Web

基于Vue

支持Hot Reload

支持远程调试

React Native

Facebook，2015年3月开源

Learn Once, Write Anywhere

支持Android，IOS

基于React

支持Hot Reload

支持远程调试

背景 - React Native 介绍

- Build native mobile apps using JavaScript and React
- 是React 的一个渲染引擎，使用Native渲染
- 外加一系列的native API
- 并提供一套JSBridge标准

工程实践

- 开发前：调试，路由，数据持久化
- 开发中：动画，控件开发
- 开发后：更新，异常监控，优化

工程实践 - 调试

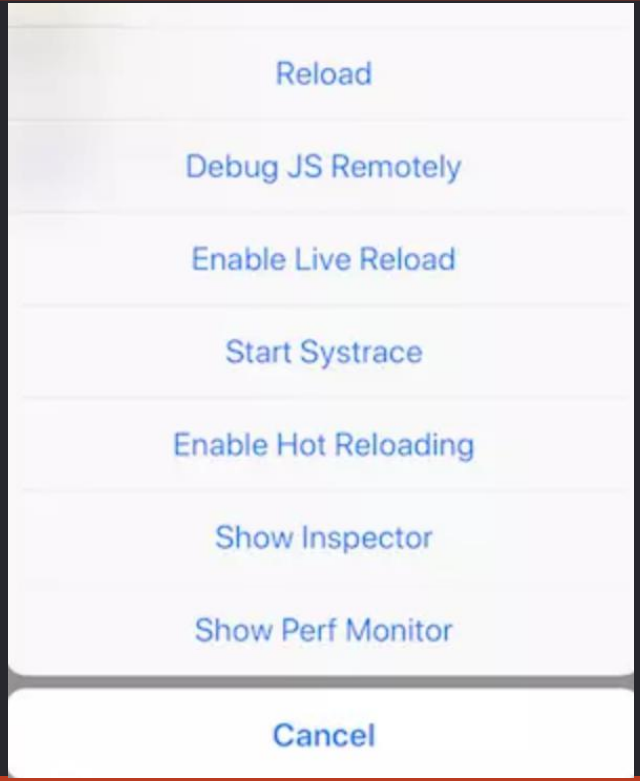
Developer Menu

打开方式：摇晃手机呼出

Reload：刷新页面，重新加载最新的JS

Enable Live Reload：自动Reload

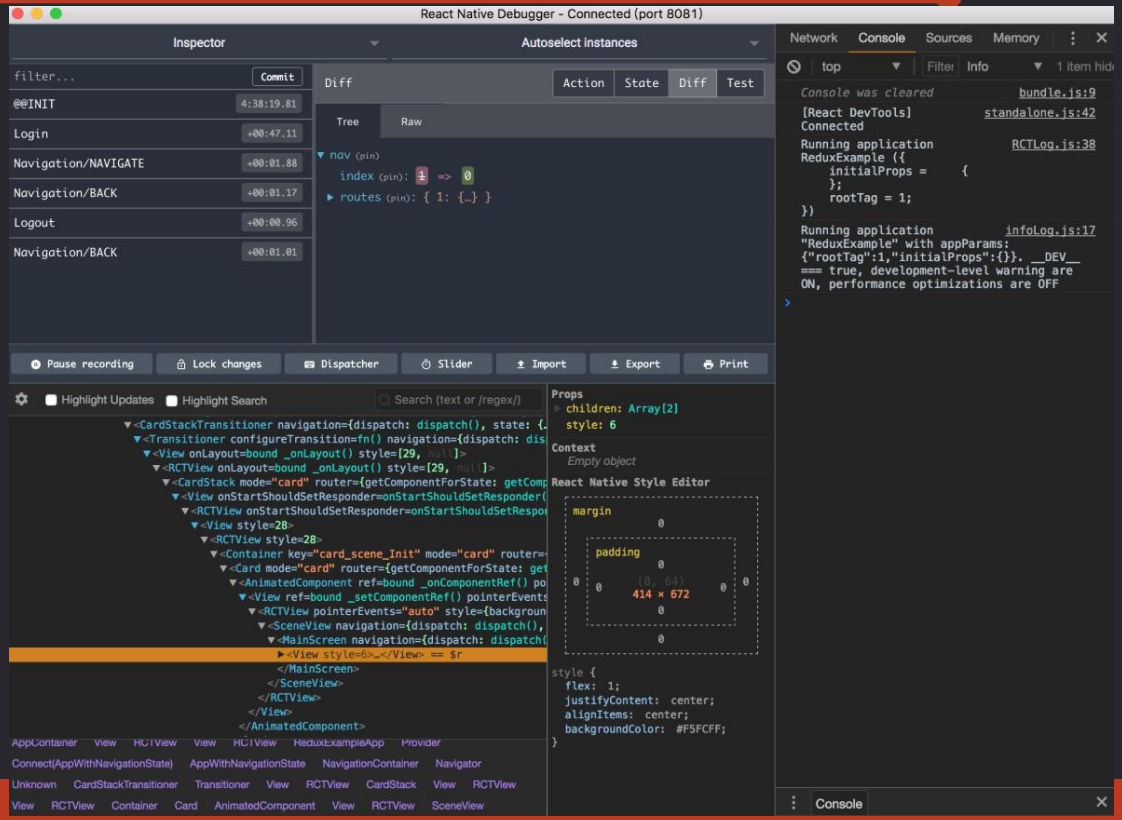
Enable Hot Reloading：代码热更新



工程实践 - 调试

React Native Debugger

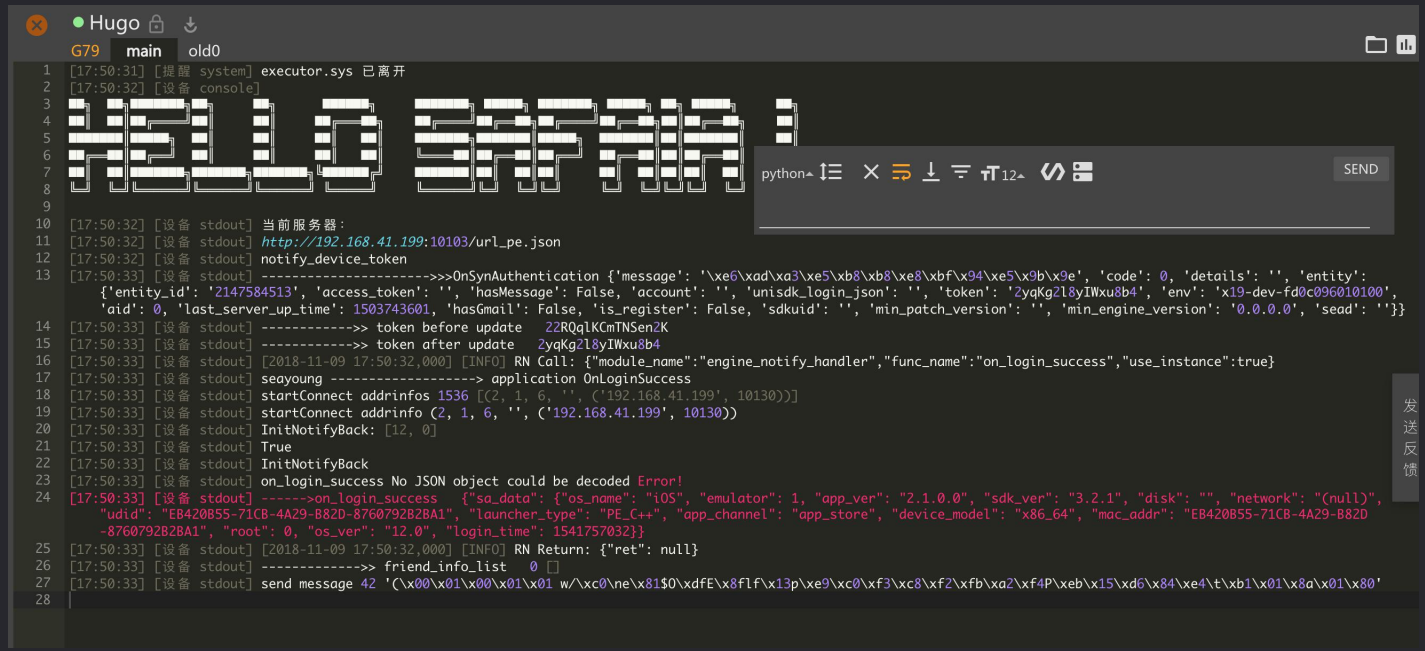
- Redux devtools
- React devtools
- Developer tools



工程实践 - 调试

Hunter

- 日志查看
- 动态插入指令



工程实践 - 路由

React Navigation

- 将路由集成到Redux中，统一管理路由
- 加入ReplaceRoute的功能
- 加入路由备份及还原的功能
- 优化连点相同路由跳转多次的问题
- 优化快速点击不同路由跳转的问题

工程实践 - 数据持久化

使用redux-persist 进行数据持久化存储

1. 通过订阅store，一旦store发生变化，就将store数据保存起来

2. 支持配置白名单和黑名单

3. 与Immutable和React Navigation一起使用时会出现异常，需要对数据额外处理

工程实践 - 动画

Animated：用于创建更精细的交互控制的动画

- 三种组件类型：View，Text，Image
- 两个值方法：Value（一维），ValueXY（二维）
- 三个动画方法：spring（弹跳），decay（衰减），timing（时间）
- 组合动画：parallel（同时执行），sequence（顺序执行），stagger（错峰执行），delay（延迟执行）
- 插值函数
- Animated.event：允许手势或其他事件直接绑定到动态值上

工程实践 - 动画

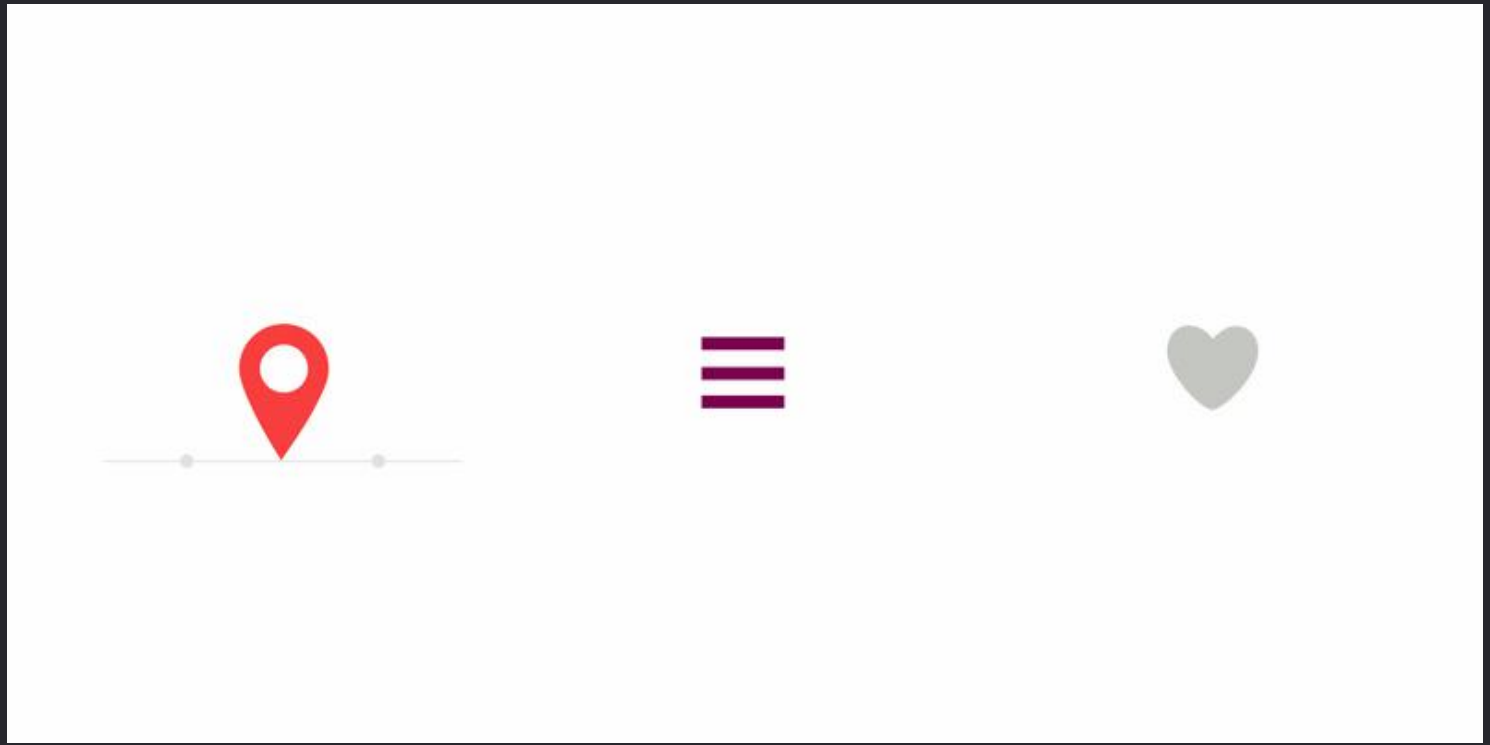
Lottie-React-Native

1. 使用Adobe After Effects软件做出特效动画

2. 通过bodymovin项目工具把特效动画采用JSON格式文件进行导出

3. 用Lottie库进行解析JSON文件并且在移动端上面渲染效果

工程实践 – **动画**



工程实践 - 动画

Lottie-React-Native

1. 节省包大小
2. 节省程序员时间
3. 高效

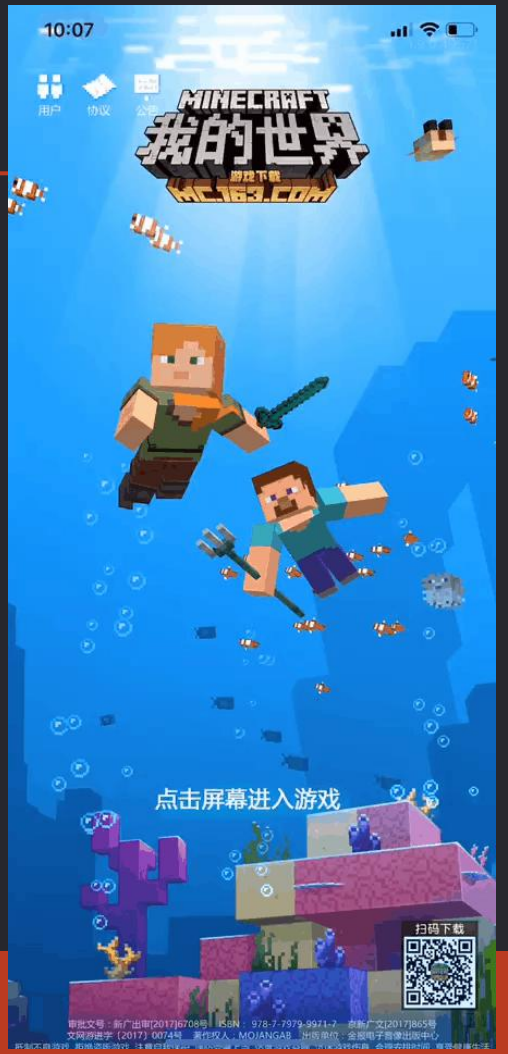
工程实践 - 动画

WebGL

WebGL简单说就是OpenGL在浏览器端的实现。

那OpenGL又是什么？

OpenGL就是一组提供了生成2d、3d图形的API。



工程实践 - 控件开发

FlatList：内存问题，不支持自定义UI，滑动bug

↓

RefreshList：自定义上拉下拉UI，内存稳定，API兼容

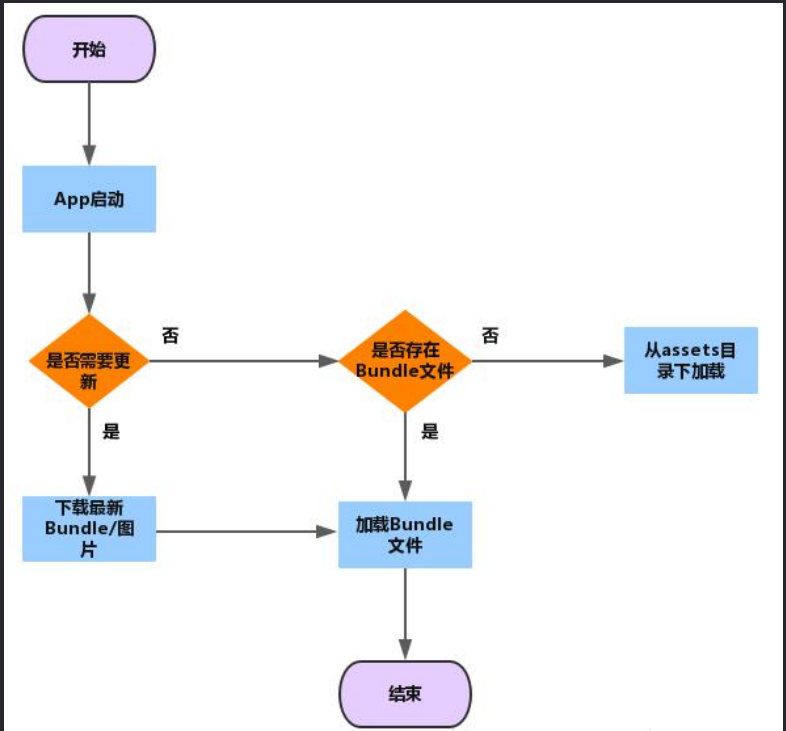
工程实践 – 更新

- 需要应对频繁升级的需求，测试及发布
- 满足同时进行多渠道多版本的更新需求
- 完善的更新策略

工程实践 – 更新

- 打包系统，打Patch系统，快速更新迭代
- 不同整包的分支管理
- 对更新出现的所有异常进行处理

工程实践 - 更新



- 检查存储空间是否足够
- 检查更新是否下载成功
- 检查资源md5是否正确
- 解压资源是否成功
- 资源是否更新成功

工程实践 - 异常监控

react-native-exception-handler

```
// js error
global.ErrorUtils.setGlobalHandler(() => {…
})
console.error = (message, error) => global.ErrorUtils.reportError(error)

// native error
ReactNativeExceptionHandler.setHandlerforNativeException(() => {…
})
```

工程实践 - 异常监控

- Sentry，Bugly（开源方案）
- App Dump（内部方案）

‹ 1 /3462 › 10 20 100 200

序号	Issue Hash	错误类型	首次crash时间	最后crash时间	Crash数	用户数	标签
1	cffe5011185a8483b9c738d9521cc7d4 1.8.0.47903	ANDROID_NATIVE_ERROR	2018-11-08 20:13:36	2018-11-09 20:11:52	4090	1573	空白 (new)
2	641fc1550feef930aa5cacafe640a1ca 1.8.3.48035(unknown)	SCRIPT_ERROR	2018-11-08 20:20:44	2018-11-09 19:43:23	2331	7	#20383
3	edab36df956b1e4daa90b89badef778f 1.8.0.47903	ANDROID_NATIVE_ERROR	2018-11-08 20:13:29	2018-11-09 20:12:05	2106	1966	libhwui.so 无意义
4	3bea9fd90176cf61d18397c58ab5f210 1.8.0.47903	SCRIPT_ERROR	2018-11-09 10:16:03	2018-11-09 20:11:54	2079	12	#20383
5	8936de9fd837df306b0b6e55d77ac304 1.8.3.48035(unknown)	SCRIPT_ERROR	2018-11-08 20:52:00	2018-11-09 19:55:44	1992	9	#20383

工程实践 - 页面响应速度优化

现象：当即将要渲染的页面，里面渲染的元素较多时，点击响应有延迟

原因：React Navigation会先将跳转的页面渲染好，再执行转场动画

解决方案：减少首屏需要加载的内容，首屏无关内容延后加载

工程实践 - 性能优化

- PureComponent，ShouldComponentUpdate，Immutable
- 减少View层级
- 去掉console
- 动画，useNativeDriver，setNativeProps
- 数据缓存，图片缓存
- Bundle拆包

工程实践 - JS Core的Bug

- iOS: JS Core
- Android: JS Core
- 调试模式： Chrome V8

Android 不支持 Symbol

Immutable里面使用到了Symbol，导致出现了一些诡异的问题

加入Symbol的polyfill解决问题

总结与思考 - React Native 的优点

- **跨平台** 使用 React Native 的大多数功能都可以实现 95 - 100% 的共享代码，和 0.2% 不同平台需要用的的文件（.android.js/.ios.js）
- **React** 最受欢迎的Web开发框架之一，使用简单功能强大
- **迭代速度** 支持热加载和热更新，效率不及H5开发，但远高于原生开发
- **性能** 大多数功能能够像原生应用一样流畅。出现性能问题时，大多数也是由于过度的渲染引起，可以通过特定属性进行优化。
- **动画** 基于RN的Animated，我们能够实现各种流畅的动画
- **JS/React 开源库** 能够使用大量的React开源库
- **Flexbox** 使用Yoga来处理布局，实现跨平台的布局渲染，用法与Web相似，能满足大部分需求

总结与思考 - React Native 的缺点

还不成熟	相对 Android 或 iOS 来说，略显不够成熟，框架比较新，迭代速度非常快，频繁升级。
JavaScriptCore不一致	Android不带有JavaScriptCore，由RN提供，版本较旧，所以容易遇到一些兼容性问题，不容易调试
开源库质量一般	太少人能精通所有平台，也导致了Android或IOS上的不一致或意想不到的错误
初始渲染较慢	渲染 React Native 需要至少一个完整的主线程 -> JS -> Yoga 布局线程 -> 主线程返回之前，然后才有足够的信息来第一次渲染屏幕
手势	Android和IOS的手势不好统一，因此在一些复杂手势上会有问题
长列表	目前已有FlatLIst对长列表进行优化，但依然远不如原生的成熟和灵活，无法兼顾性能和流畅度
稀奇古怪的崩溃	依然有不少难以解决、非常奇怪的崩溃存在，不好复现。

Thank you