SDCC 2016
中国软件开发者大会
SOFTWARE DEVELOPER CONFERENCE CHINA

全民K歌React Native实践与优化

袁聪（eddyyuan）

目录

1 React Native简介

2 React Native原理

3 React Native优化

4 展望

目录

1 React Native简介

2 React Native原理

3 React Native优化

4 展望

React Native简介

- 基于React开发
- 跨平台特性
- 原生Native体验
- 热更新

React Native简介

H5	RN	Native
易传播	性能较好	性能好
性能差	用户体验较好	用户体验好
用户体验差	开发周期短	开发周期长
开发周期短	更新成本较低	更新成本高
更新成本低	维护成本较低	维护成本高
维护成本低		

目录

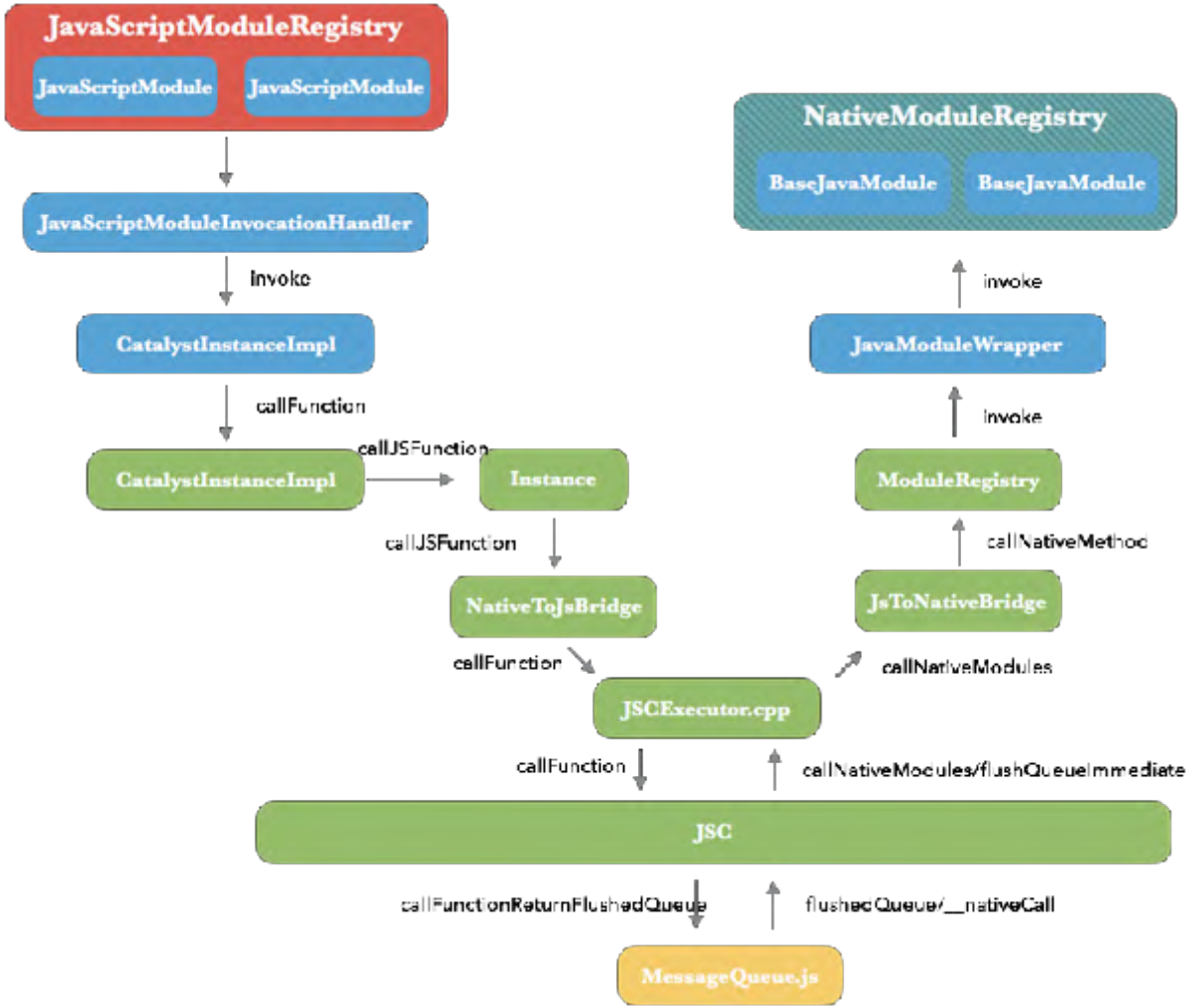
1 React Native简介

2 React Native原理

3 React Native优化

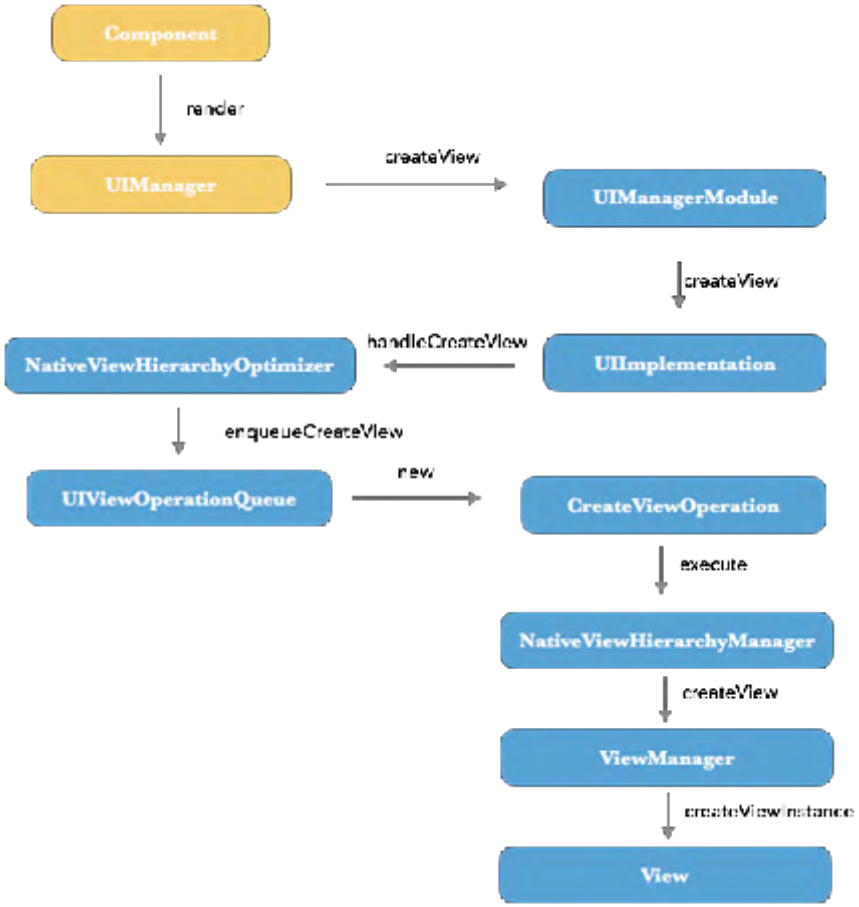
4 展望

React Native原理——通信机制



React Native原理——UI渲染

怎样从JS代码中产生一个Native的View控件呢？



目录

1 React Native简介

2 React Native原理

3 **React Native优化**

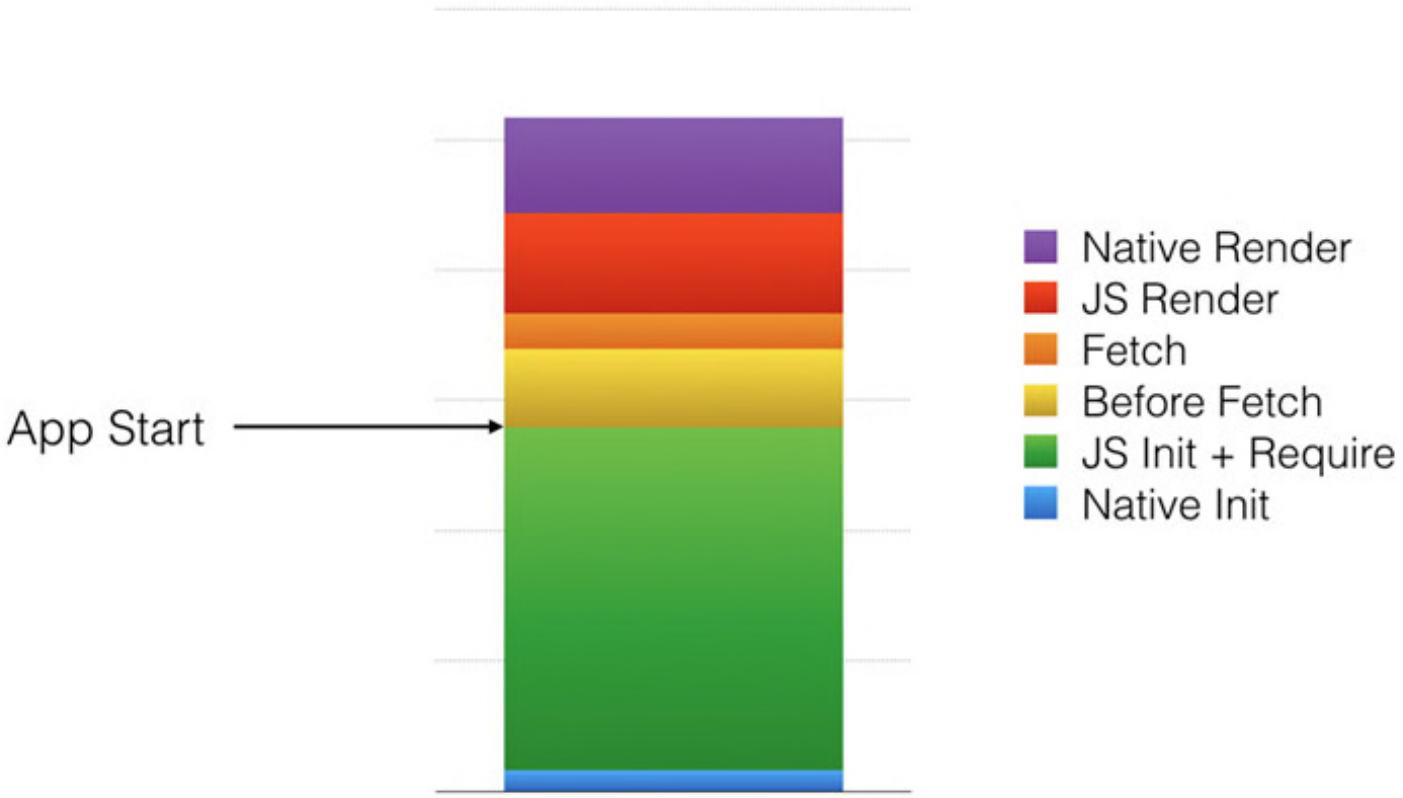
4 展望

React Native优化——Bundle拆分

- 业务分离，按需加载，减少资源消耗
- 避免执行大量JavaScript代码带来的性能问题
- 更灵活的优化策略
- 减少更新时的流量消耗

React Native优化——Bundle拆分

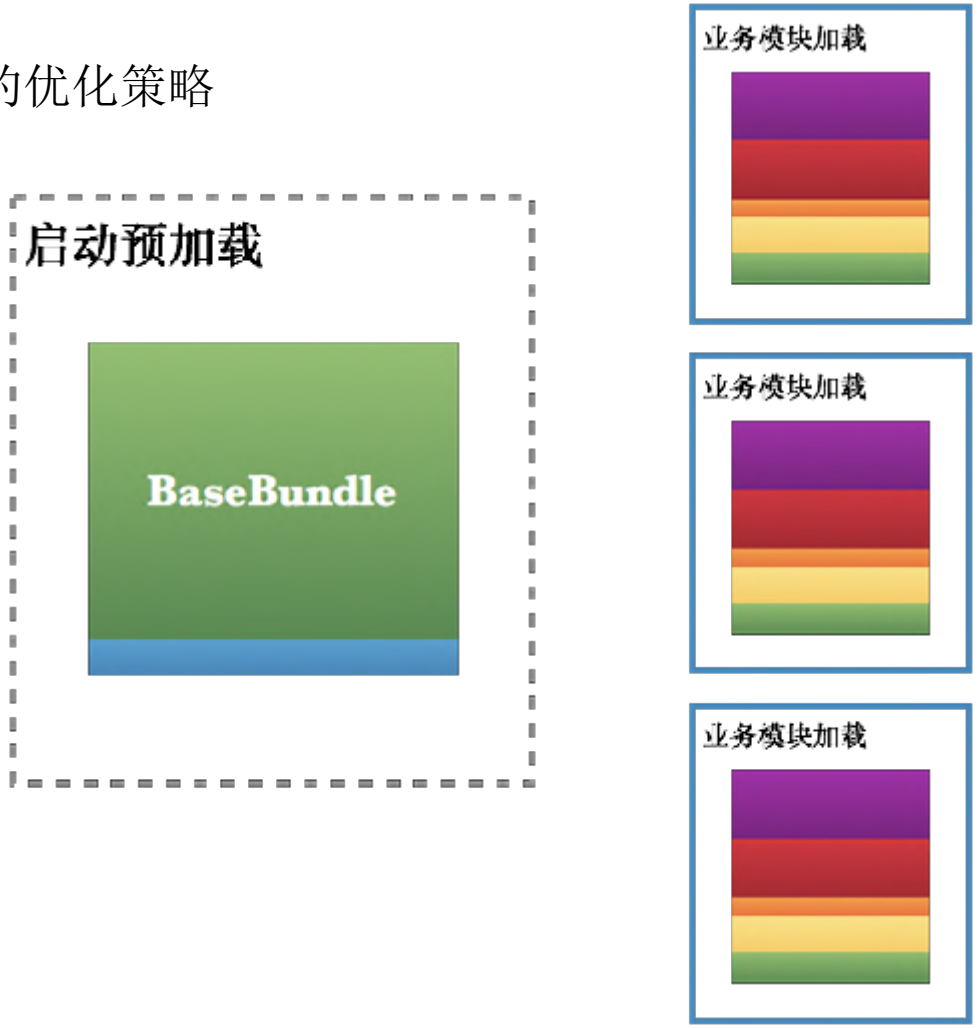
启动的性能瓶颈在于JavaScript的执行



图片来源：Dive into React Native Performance

React Native优化——Bundle拆分

更灵活的优化策略



React Native优化——Bundle拆分

减少更新时的流量消耗

React Native优化——Bundle拆分

FB的unbundle方案（开发中）：

- Android：将模块拆分成巨量的单独的小文件，需要时NativeRequire加载
  （可优化点：小文件合并）

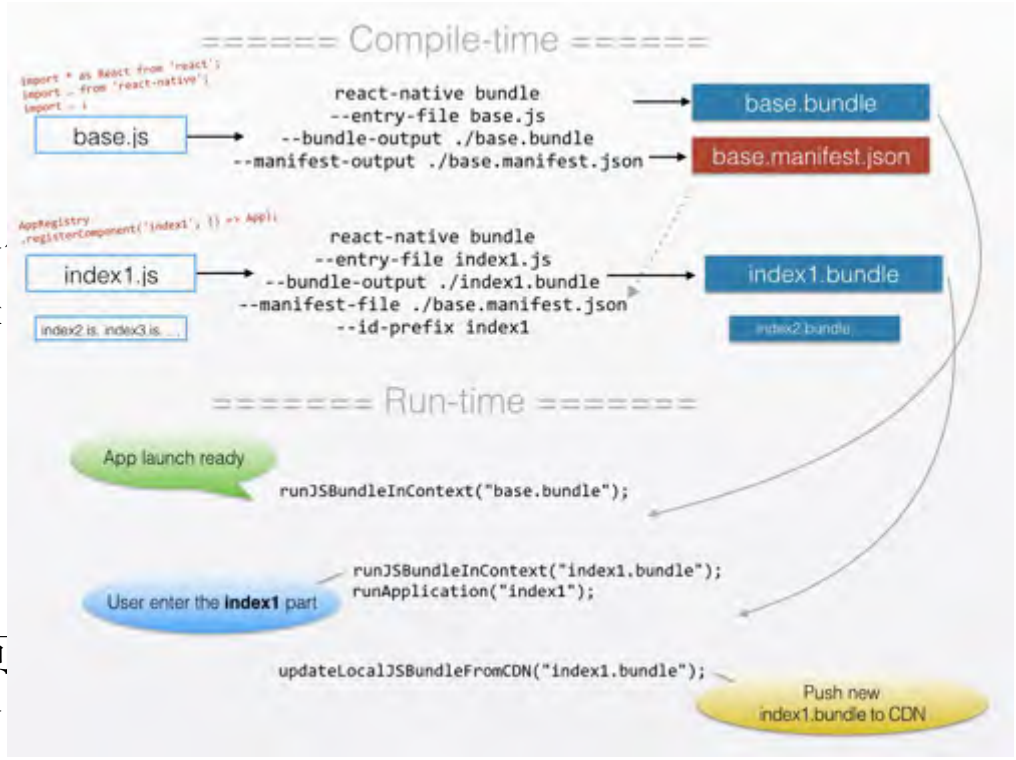
名称	修改日期	大小
UNBUNDLE	今天 下午2:19	4 字节
547.js	今天 下午2:19	739 字节
546.js	今天 下午2:19	4 KB
545.js	今天 下午2:19	2 KB
544.js	今天 下午2:19	1 KB
543.js	今天 下午2:19	612 字节
542.js	今天 下午2:19	458 字节
541.js	今天 下午2:19	309 字节
540.js	今天 下午2:19	2 KB
539.js	今天 下午2:19	885 字节
538.js	今天 下午2:19	2 KB
537.js	今天 下午2:19	559 字节
536.js	今天 下午2:19	6 KB
535.js	今天 下午2:19	3 KB

- iOS：单独Bundle文件（避免I/O性能问题，Android表示哭晕在厕所），
  在文件头写入索引，module通过null byte分隔，需要时根据索引加载

React Native优化——Bundle拆分

K歌的方案：

- 打包出基础包（包含RN核心、
- 导出包含的模块列表
- 根据基础包模块列表打包业务
- 过滤业务包的依赖和重复模块



优势：

- 按业务拆分
- 无较多小文件导致的I/O性能问
- 分包粒度灵活，大到整个业务

React Native优化——动态加载

React Native + MultiDex + 动态加载能碰撞出什么火花呢？

突破React Native热更新局限性，无限扩展

React Native热更新局限性：

React Native热更新是基于已有Native Module和View的热更新

React Native优化——动态加载

方案一：构建一个通用的代理NativeModule，传递需要执行的类名、方法名以及参数列表，通过反射去调用

方案二：去了解NativeModule的注册和调用流程，动态向其中添加新的Native组件

优劣：

方案一简单易于实现，但是抛弃RN已有架构，需自己维护对象和构建参数，同时对JS非透明，应用Native版本升级时需要兼容不同的调用方式，版本控制比较麻烦。

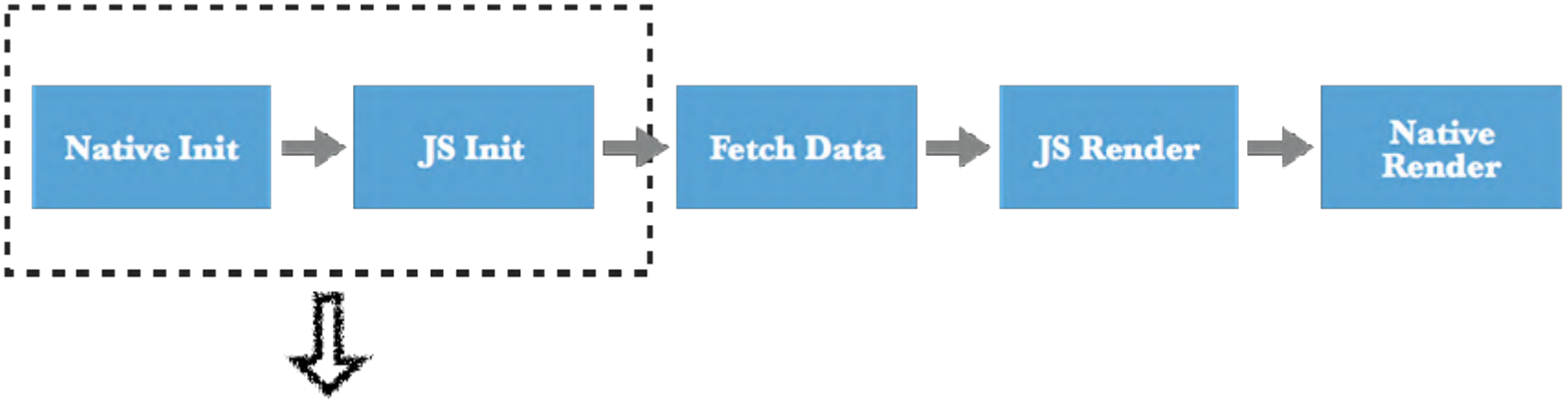
方案二实现难度稍大，沿用RN的架构，对JS透明，应用Native版本升级可以随之合并进最新APK中

React Native优化——动态加载

Native Module动态加载：

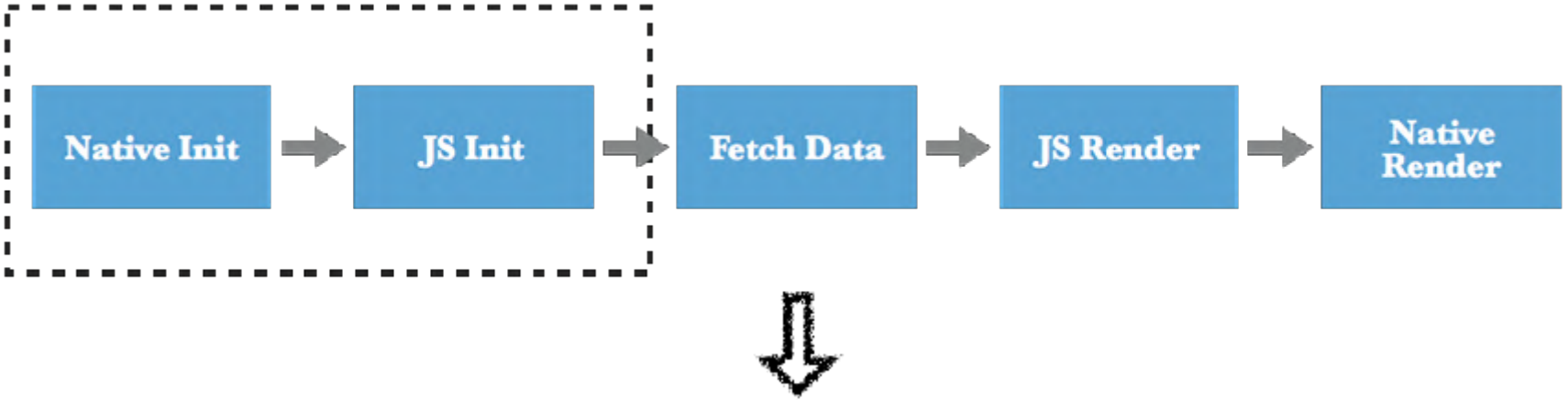
① 在ModuleRegistry的modules_变量中增加新的NativeModule
② 调用moduleNames更新modulesByName_同时得到moduleID
③ 向JS中MessageQueue对象的RemoteModules中添加新的NativeModule描述
④ 向JS中的NativeModules对象添加新的NativeModule描述

React Native优化——首屏秒开



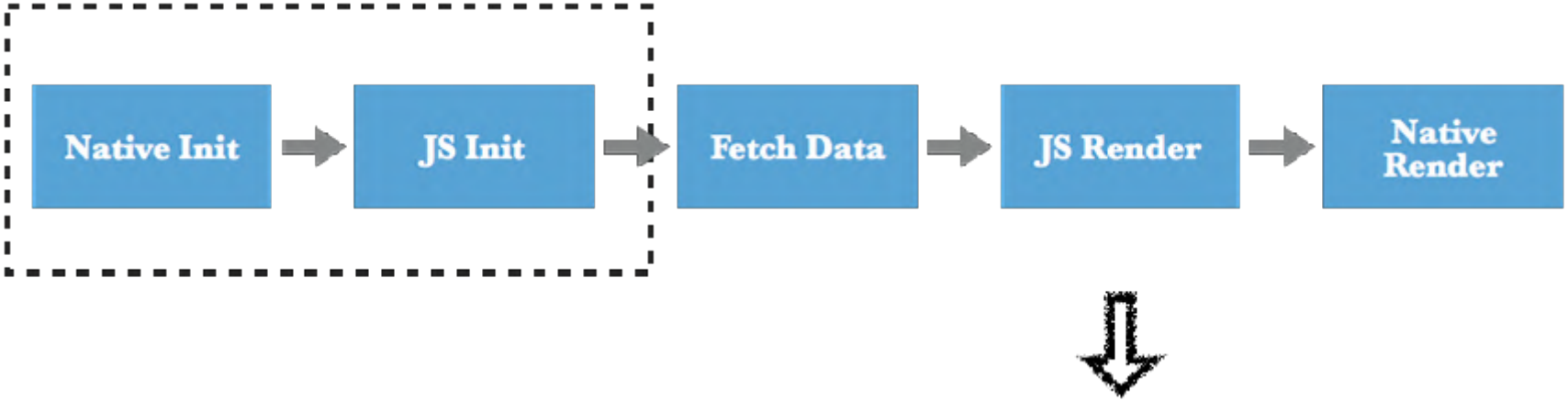
- ReactInstanceManager单例化
- ReactContext预加载
- 拆分Bundle，减少JS执行时间
- 减少JS Module注册
- 删除JS中_DEV_代码

React Native优化——首屏秒开



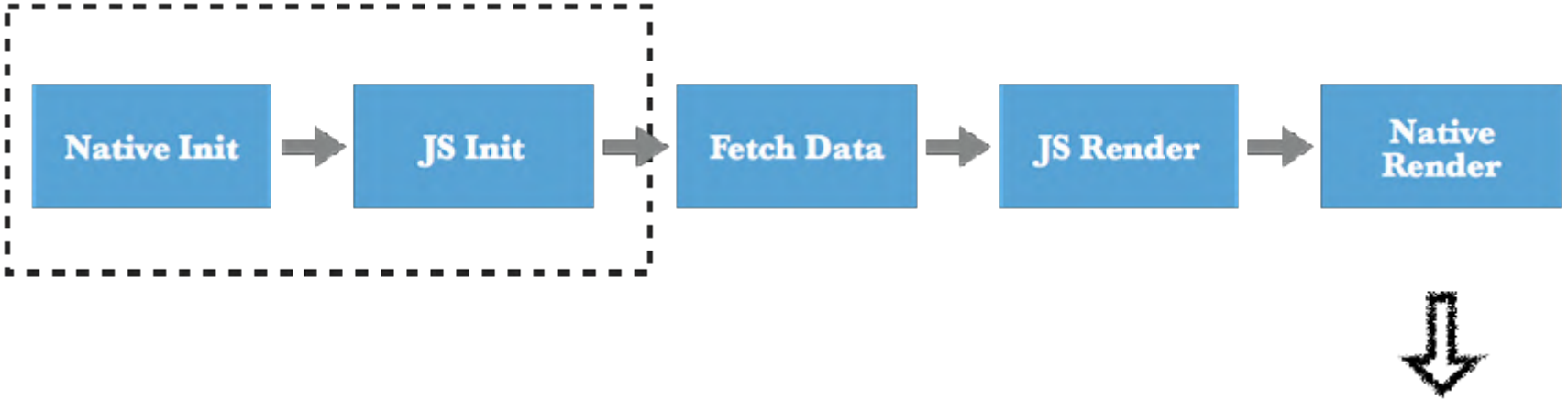
- 优先加载本地缓存
- 常用图标图片本地打包

React Native优化——首屏秒开



- 减少不必要的动画
- 合并Native调用，降低调用频率
- 使用客户端执行性能高的代码

React Native优化——首屏秒开



- 布局层级优化

React Native优化——首屏秒开

布局层级优化

- JS层减少嵌套以及不使用会造成多层布局的属性
- 源码修改减少层级
- Native通过自定义View的形式实现复杂布局提供给JS调用

React Native优化——首屏秒开

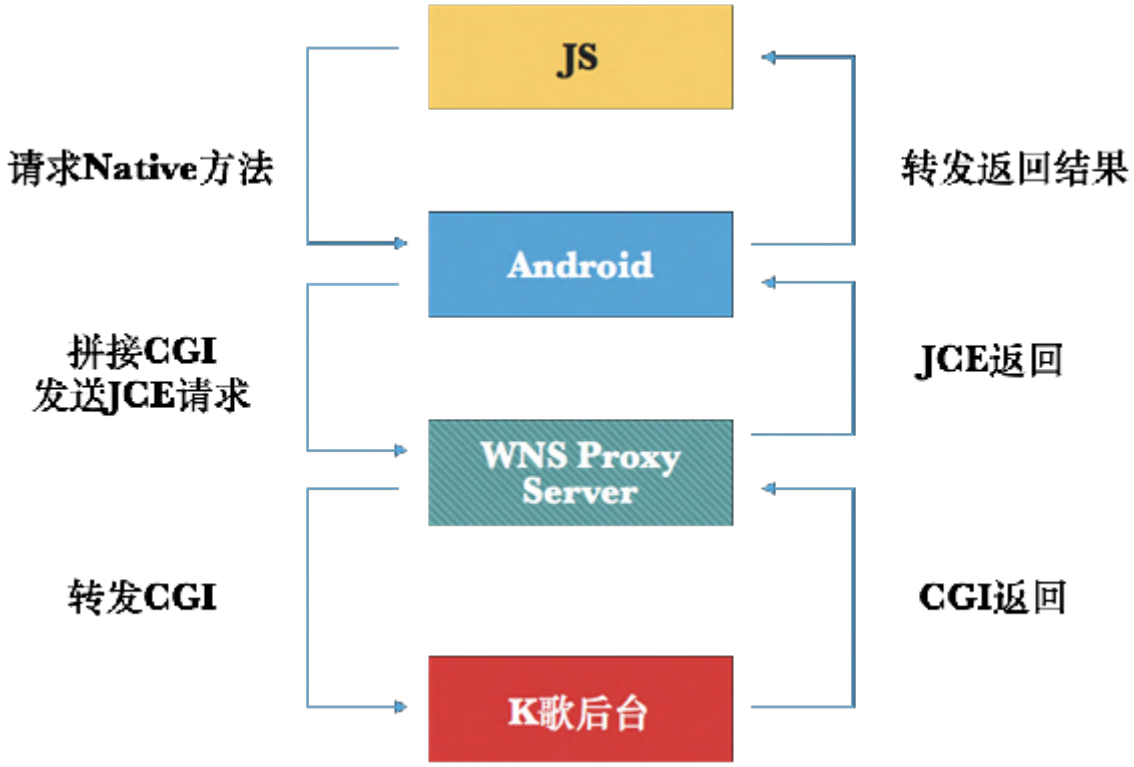
数据对比：

Android（Note3）						
	RN	H5	RN	H5	RN	H5
Wifi+Cache	483.2	/	511.5	/	538.1	/
WIFI	954.7	3019.6	973.1	2936.5	1047.3	3362.7

React Native优化——网络优化

DNS解析慢
DNS劫持
连接速度慢
弱网络兼容差
……

IP直连
竞速和就近接入
长连接
更小的请求包
重试机制
……



React Native优化——其他优化

内存：

排查泄露，一一处理，耗时费力（复杂）
RN单独进程实现，不需要时结束进程（相对简单，但占资源，进程通信复杂）

安装包大小：

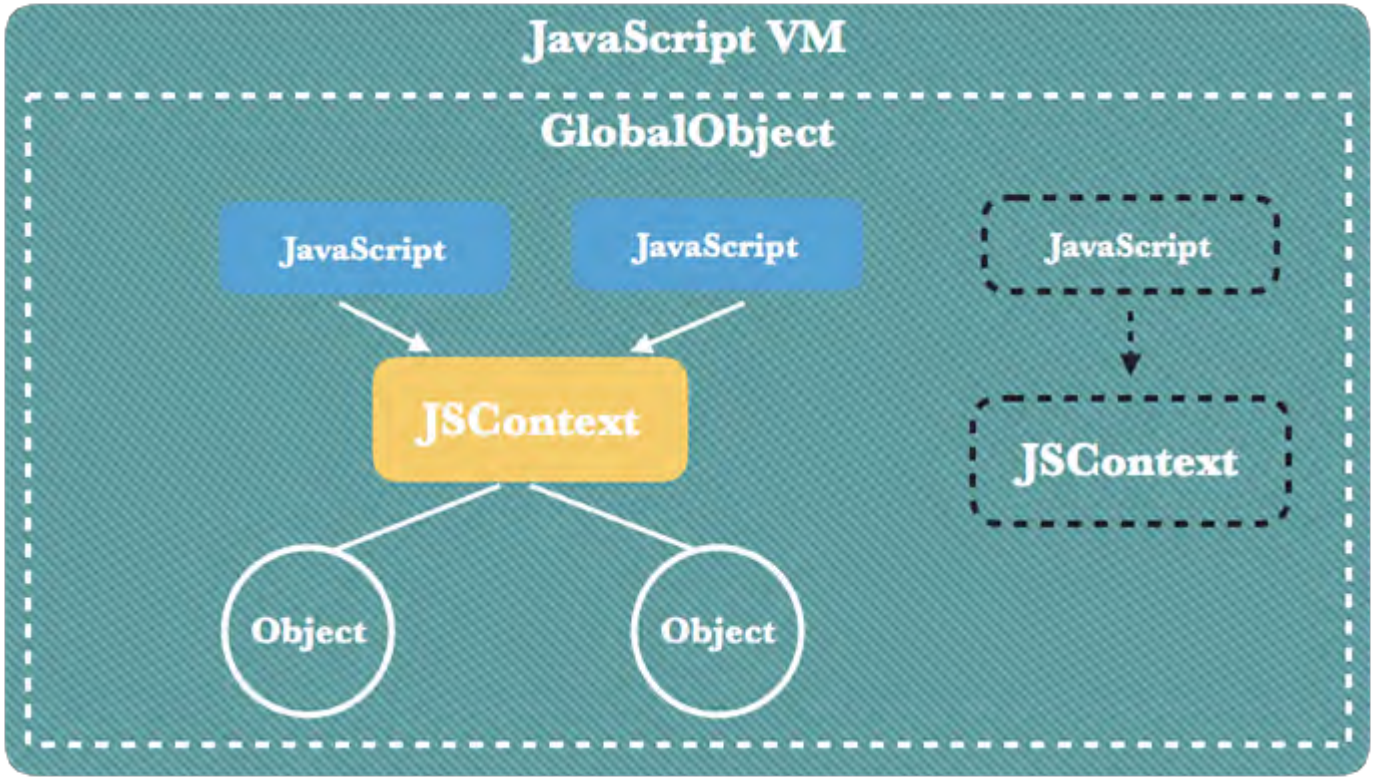
SO下载并手动加载（简单，需注意SO依赖关系）
插件化（复杂）

安全性：

文件合法性校验

React Native优化——后续优化方向

- 差分算法增量更新
- Hot Reload



目录

1 React Native简介

2 React Native原理

3 React Native优化

4 展望

展望

- 和Native更小的性能差异
- 更丰富、更高效的组件和动画
- 可能成为中小微型体量应用的首选开发方案
- 可能替代大部分H5的应用场景

SDCC 2016

中国软件开发者大会

SOFTWARE DEVELOPER CONFERENCE CHINA

谢 谢！