

React Native在携程的工程实践

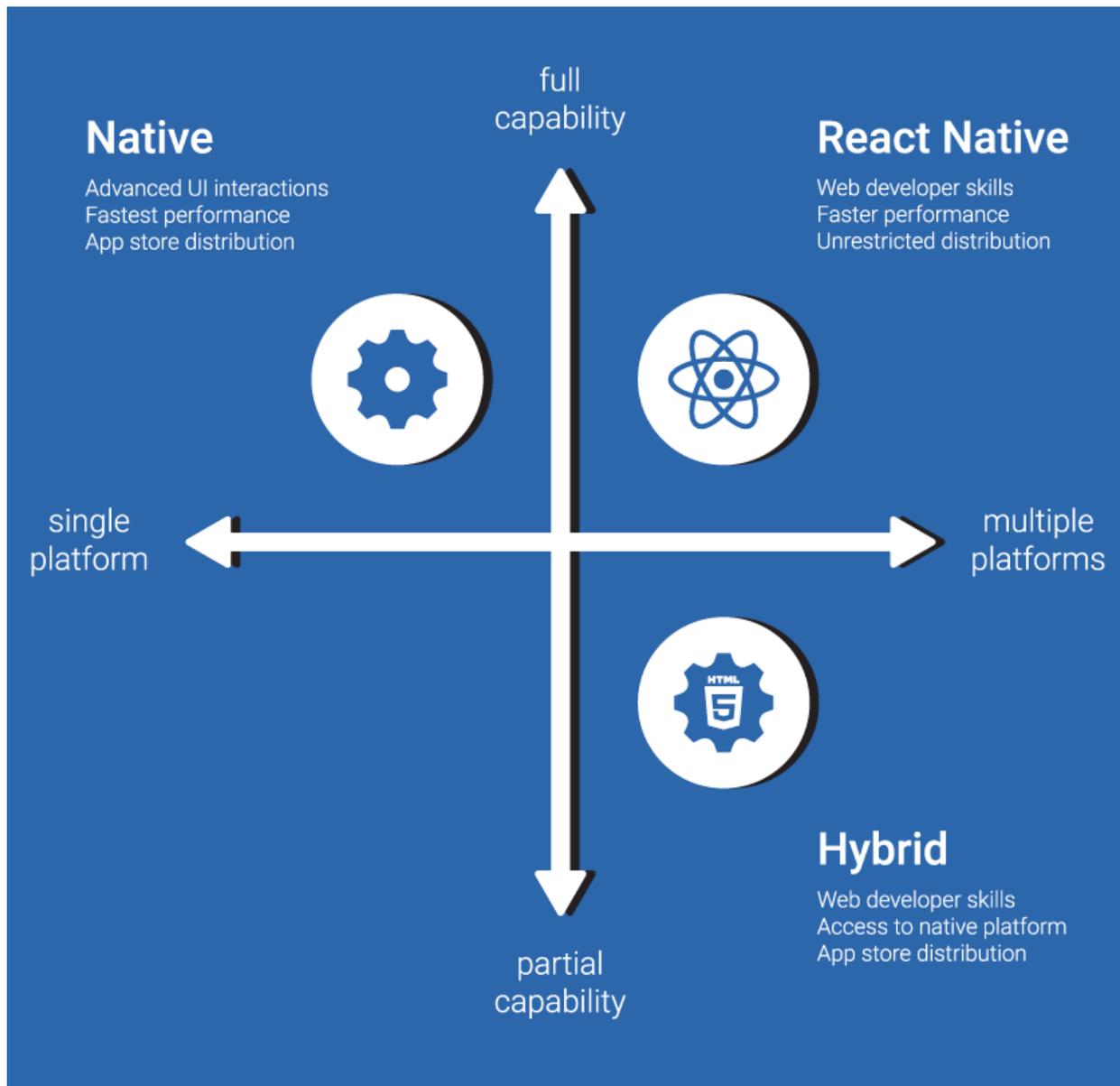
廖利强

携程 - 技术研发中心 - 基础业务研发部

目录

1. React Native在携程的发展与现状
2. Ctrip React Native框架平台介绍
3. 携程React Native工程实践经验分享
4. 总结与思考

ReactNative在携程的发展——Why



ReactNative在携程的现状——When&What



2017.01



规模化使用

升级RN0.41版本，相关配套系统逐步完善，业务大量接入

2016.08



性能优化

升级RN0.30版本，完成包拆分和后台预加载等核心性能问题

2016.04



调研引入

组件API相对稳定，0.28-stable版本入手研究

2015.10



探索阶段

RN Android版本开源之后，业务团队尝试，探讨

2015.05



体验阶段

iOS App V6.5版本发票抬头页面使用RN开发



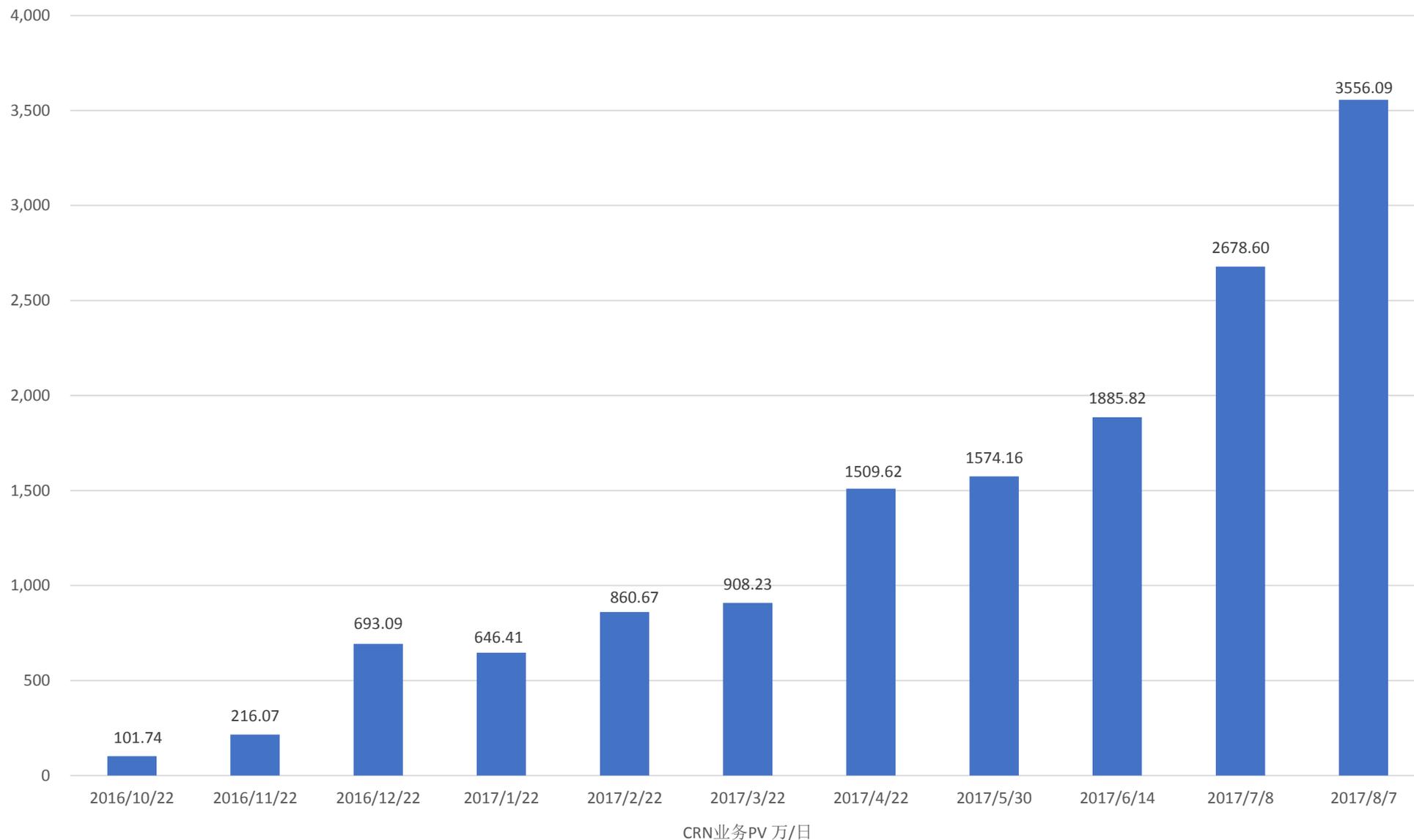
ReactNative在携程的现状——Statistic



76
单业务最多页面数
业务复杂度高

377
总共业务页面数量
业务大量使用

45
业务模块数
所有团队都在使用



CRN框架介绍——What

CRN是由无线基础研发团队基于React Native开发的适用于携程App的框架平台：

- 支持iOS / Android双平台；
- 服务于携程业务并提供全生命周期开发支持；
- 作用于加快开发效率、降低开发成本、提升运行性能、增强APP质量和灵活性；



CRN框架介绍——How



需求解决

采用协作模式，业务团队负责业务功能开发，框架团队负责框架开发、基础设施以及整体把控

业务
开发

后期处理

提供完善的异常和性能监控，解决存在于用户使用过程中的问题痛点

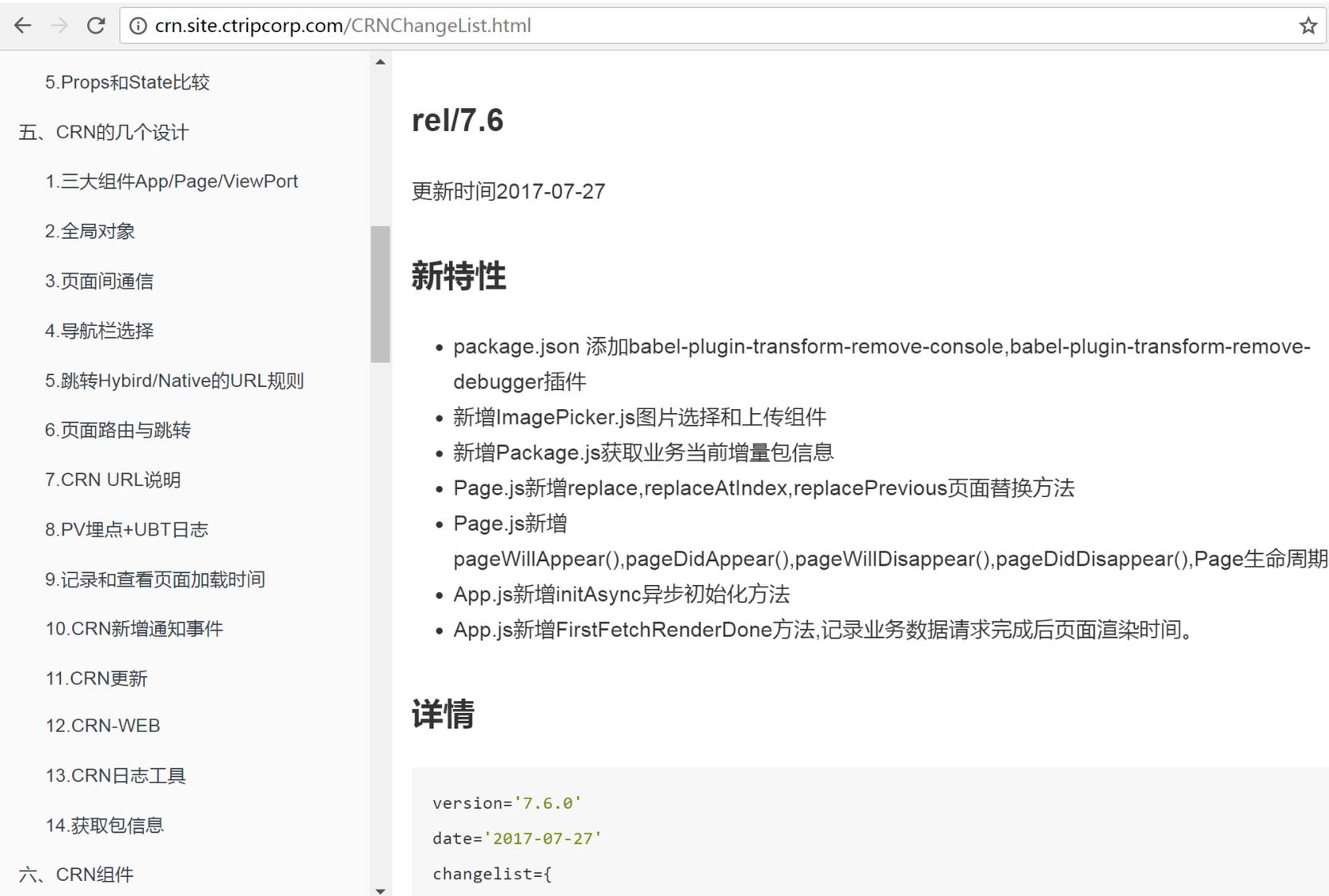
性能
监控

框架
开发

发布方案

采用携程无线持续交付平台MCD统一发布，使用crn-packer工具分拆common和business打包，区分FAT、UAT、PRD三环境增量发布

MCD
发布



5. Props和State比较

五、CRN的几个设计

- 1. 三大组件App/Page/ViewPort
- 2. 全局对象
- 3. 页面间通信
- 4. 导航栏选择
- 5. 跳转Hybird/Native的URL规则
- 6. 页面路由与跳转
- 7. CRN URL说明
- 8. PV埋点+UBT日志
- 9. 记录和查看页面加载时间
- 10. CRN新增通知事件
- 11. CRN更新
- 12. CRN-WEB
- 13. CRN日志工具
- 14. 获取包信息

六、CRN组件

rel/7.6

更新时间2017-07-27

新特性

- package.json 添加babel-plugin-transform-remove-console, babel-plugin-transform-remove-debugger插件
- 新增ImagePicker.js图片选择和上传组件
- 新增Package.js获取业务当前增量包信息
- Page.js新增replace, replaceAtIndex, replacePrevious页面替换方法
- Page.js新增
pageWillAppear(), pageDidAppear(), pageWillDisappear(), pageDidDisappear(), Page生命周期
- App.js新增initAsync异步初始化方法
- App.js新增FirstFetchRenderDone方法, 记录业务数据请求完成后页面渲染时间。

详情

```
version='7.6.0'  
date='2017-07-27'  
changelist={
```

 CRN介绍

 RN重点概念介绍

 CRN快速上手

 CRN设计理念

 CRN组件介绍

 CRN发布监控

 FAQ

API定义

类 模块

Type to filter APIs

ABTesting
AddressBook
App
Application
BirthdayPicker
Bridge
Business
Button
Calendar
Call
Channel
CountryCode
CRNDev
CRNListView
CRNListViewDataSource
DatePicker
DatePickerWidget
Device
Encrypt
Env
Event
Fetch

App Class

Show: Inherited Protected Private Deprecated

定义在: [lib/App.js:37](#)

模块: [UI](#)

可用版本 $\geq 6.19.0$

App - CRN app 入口

App 是一个组件，也是一个基类。

```
import {Component} from 'react';

class App extends Component {
}
```

每一个使用使用 `AppRegistry.registerComponent` 注册的组件都应该是一个 App 子类。改模块加载时会 new 出一个实例，我把这个实例称之为 app。（下文中出现的 app 都是指该 app 而不是 Native App）

```
Class HelloWorld extends App {
}
AppRegistry.registerComponent('HelloWorld', () => HelloWorld);
```

App 作为一个入口组件主要提供了下面几个功能：

CRN-CLI

- 工程初始化
- 启动Server
- 启动App并打开当前业务
- 远程Log日志输出
- 运行CRNExplorer工程

[crn-cli]:

```
crn-cli  init          <project-name>  建立并初始化RN工程
crn-cli  start        --port          启动RN服务,默认端口5389
crn-cli  run-ios      <term>          运行Ctrip IOS
crn-cli  run-android <term>          运行Ctrip Android
crn-cli  run-patch    执行patch,替换依赖文件
crn-cli  log          本地启动日志服务,默认端口5387
crn-cli  example     建立CRN组件和API调用示例工程
crn-cli  cli-update   更新cli版本
crn-cli  crn-update   更新@ctrip/crn版本
crn-cli  pack         <term>          本地打包
crn-cli  preBuildCheck --client-version 预编译检查,默认使用客户端最新版本
```

where <term> is below:

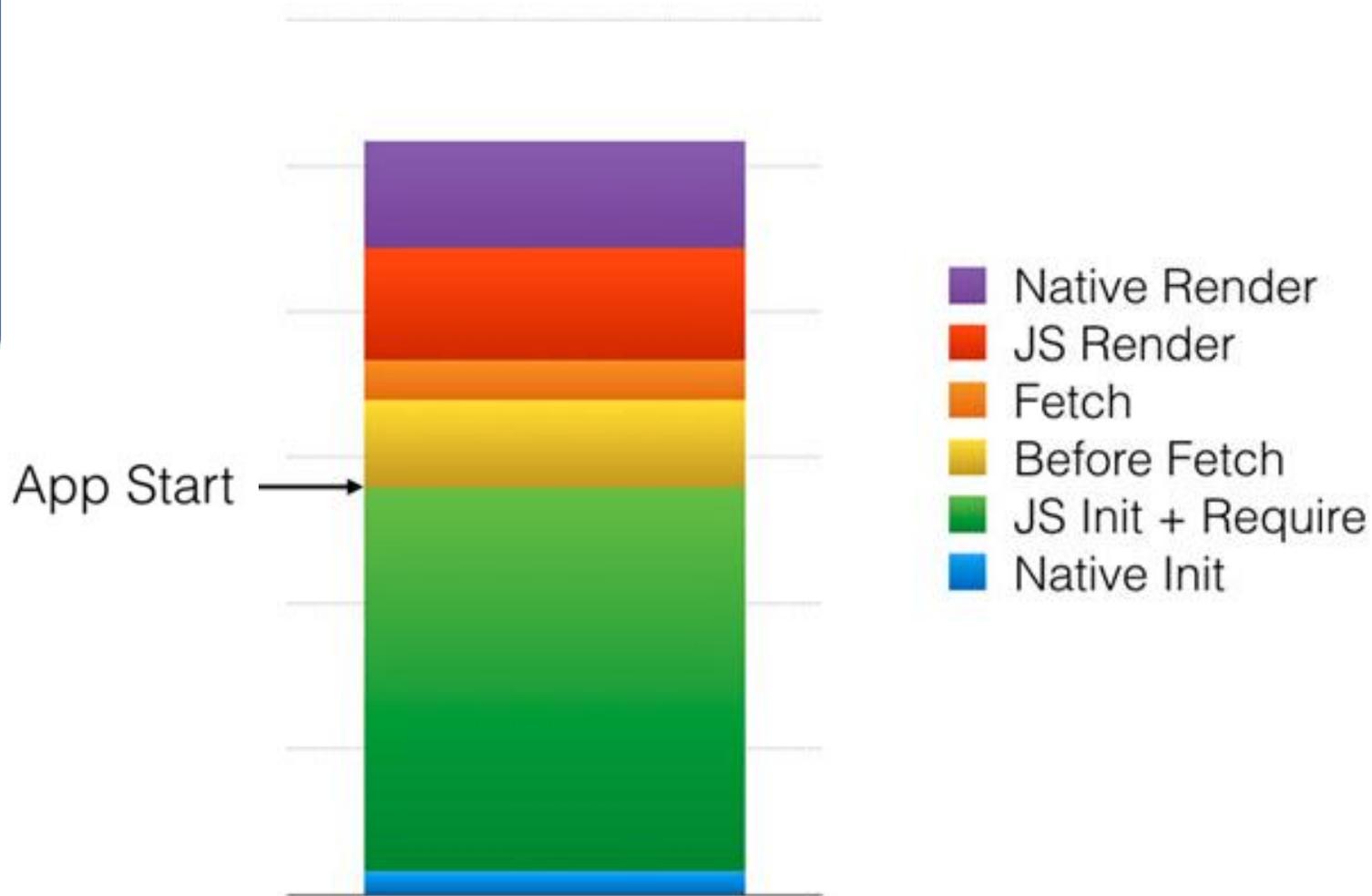
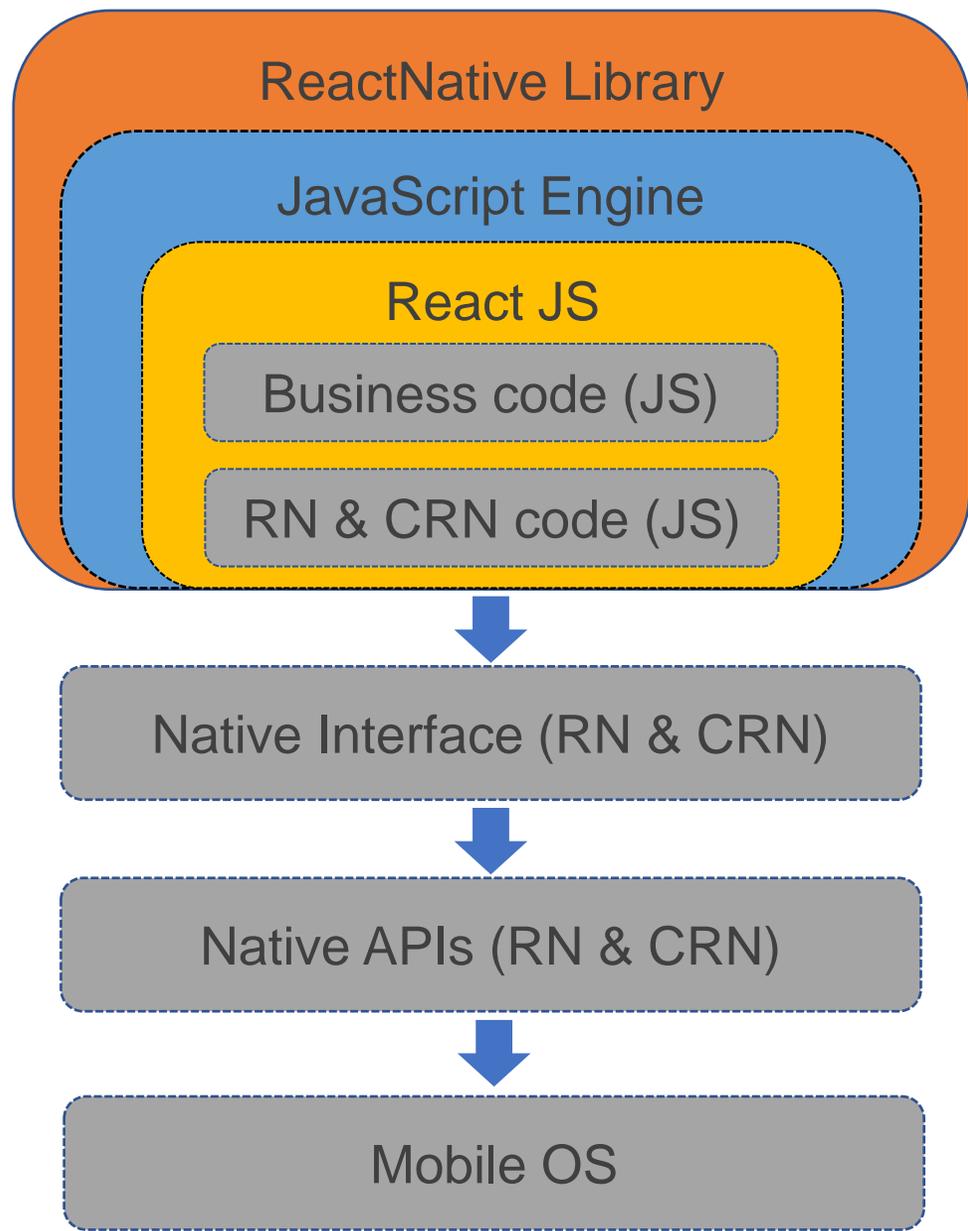
```
--ip          选填IP地址,默认本机IP
--query       选接相应参数: key1=value1+key2=value2,请使用+连接
--app-version 选填相应APP版本: 705,704,703,702,701
--reset       清除本地缓存文件
--productCode [本地打包]rn_xxx, BU MCD发布的模块对应的频道号,如站内信对应rn_message
```

<http://crn.site.ctripcorp.com/>

CRN-Packer

- 各版打包工具
- 初始化(patch、mapping)
- 打拆分包

携程RN工程实践经验——Run&Cost



携程RN工程实践经验——Bundle1



```
'use strict';

/* 1.模块引用申明部分 */
import React, { Component } from 'react';
import { AppRegistry } from 'react-native';
import { HelloWorldView } from './HelloWorldView'

/* 2.入口模块定义部分 */
class Index extends Component {
  function render() {
    return (<HelloWorldView/>)
  }
}

/* 3.入口模块注册部分 */
AppRegistry.registerComponent('RNMessage', () => Index);
```

```
/* 1. 头部--全局定义部分 */
(function(global) {
  global.__DEV__=true;
  global.__BUNDLE_START_TIME__=Date.now();
})(typeof global !== 'undefined' ? global : typeof self !== 'undefined' ? self : this);

/* 2. 中间--各模块定义部分 */
_d(0 /* RNMessage/index.android.js */, function(global, require, module, exports) {
  /*...code...*/
  module.exports=require(12 /* ./src/index */);
}, "RNMessage/index.android.js");
_d(188 /* InitializeJavaScriptAppEngine */ , function(global, require, module, exports) {
  /*...code...*/
  require(80 /* RCTDeviceEventEmitter */ );
}, "InitializeJavaScriptAppEngine");
_d(473 /* BorderRadius */ , function(global, require, module, exports) {
  /*...code...*/
  module.exports = BorderRadius;
}, "BorderBox");
_d(474 /* resolveBoxStyle */ , function(global, require, module, exports) {
  /*...code...*/
  module.exports = resolveBoxStyle;
}, "resolveBoxStyle");

/* 3. 尾部--引擎初始化+执行入口模块 */
;require(188);//InitializeJavaScriptAppEngine
;require(0);//入口模块
```

 打包前源码

 打包后代码

携程RN工程实践经验——Bundle2



```
import React, { Component } from 'react';
import {
  AppRegistry,
  View,
  DeviceEventEmitter
} from 'react-native';

/* 1. 添加全局监听 */
var mainComponent = null;

DeviceEventEmitter.removeAllListeners(); //fixed ios multi-trigger
DeviceEventEmitter.addListener("RenderModuleEventListener", function(event) {
  console.log("Render moduleId: " + event.moduleId);
  if (event && event.moduleId) {
    mainComponent = require(event.moduleId);
  }
});

/* 2. 定义FakeApp模块 */
var FakeApp = React.createClass({
  render: function() {
    var _content = null;
    if (mainComponent) {
      _content = React.createElement(mainComponent, this.props);
    }
    return _content || <View/>;
  },
});

/* 3. 注册FakeApp模块 */
AppRegistry.registerComponent('FakeApp', () => FakeApp);
```

```
{
  "path": "/crn_common_entry.js",
  "id": 0
},
{
  "path": "/node_modules/react-native/packager/react-packager/src/Resolver/polyfills/prelude.js",
  "id": 1
},
{
  "path": "/node_modules/react-native/packager/react-packager/src/Resolver/polyfills/require.js",
  "id": 2
},
{
  "path": "/node_modules/react-native/Libraries/CustomComponents/Lists/FillRateHelper.js",
  "aliasName": "FillRateHelper",
  "id": 496
},
{
  "path": "/node_modules/@ctrip/crn/lib/CRN.js",
  "id": 333333
},
{
  "path": "/node_modules/@ctrip/crn/lib/Bridge.js",
  "id": 333334
},
{
  "path": "/node_modules/@ctrip/crn/lib/Call.js",
  "id": 333335
},
{
  "path": "/node_modules/@ctrip/crn/lib/CtripSwitchBar.js",
  "id": 333418
}
```

 Fake入口模块代码

 ID与模块匹配数据

注入
ID

注入上次打包Common ID Config

- 保证业务可引用同一个common包
- 保证JS模块的前后兼容

模块图
生成

Dependency Graph Analysis

- 使用到的模块及模块总数在此决定
- 模块、模块名、路径数据在此组装

替换
模块

替换用到common的模块ID

- 保证不把重复代码打包进业务
- 保证common的模块ID在业务包里面的一致性

输出
模块

输出最终文件

- JS文件
- ID Config文件
- Source-map文件

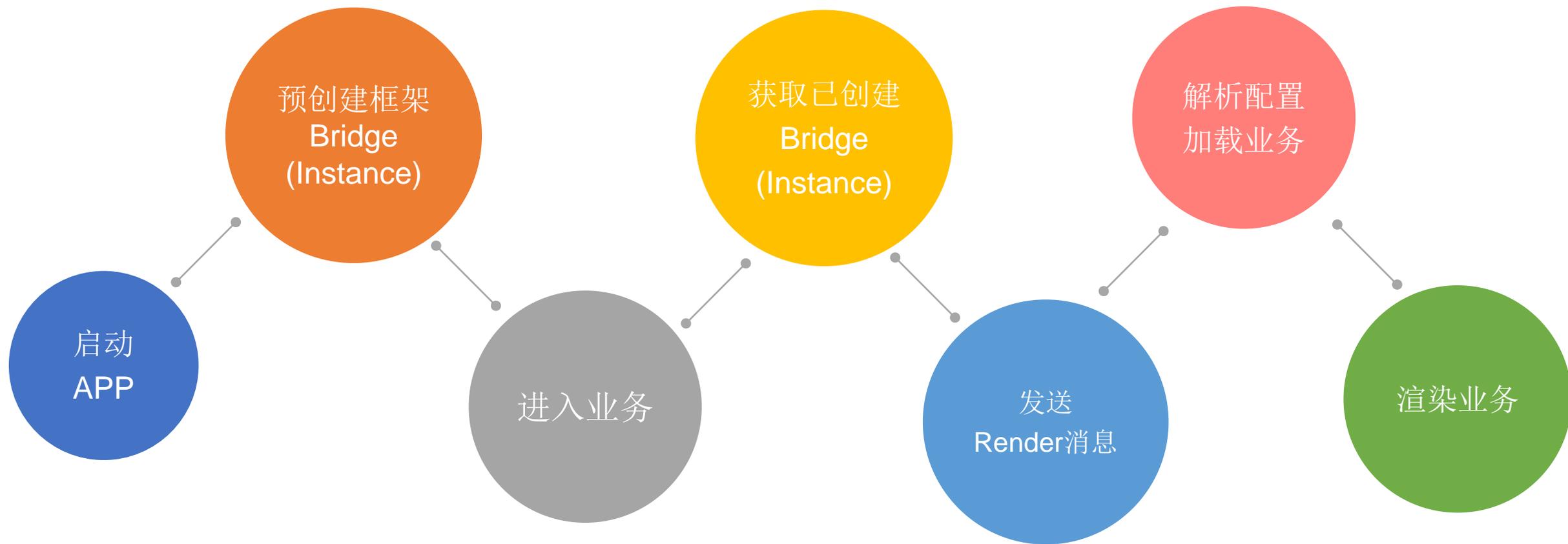
自定义
处理

添加一系列自定义操作

- JS文件合并
- 附加统计代码
- 添加包识别文件

* 罗列携程在打包过程的改造点

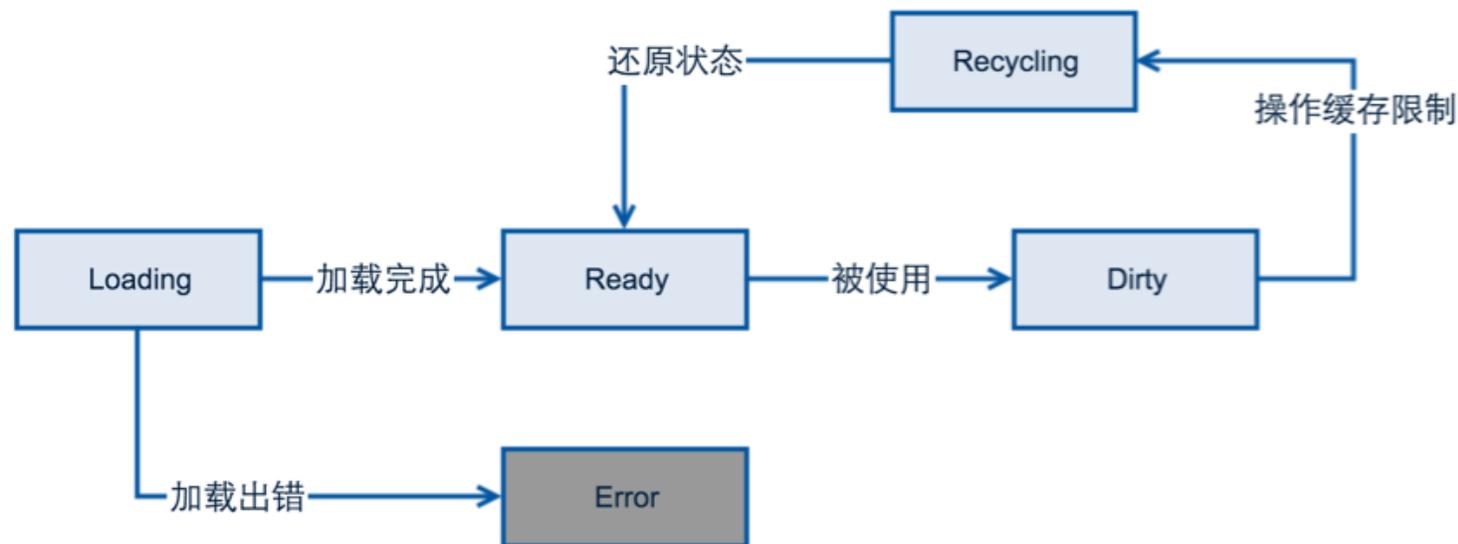
携程RN工程实践经验——Preload



* 加载或渲染业务阶段失败，则退出当前业务并提示用户

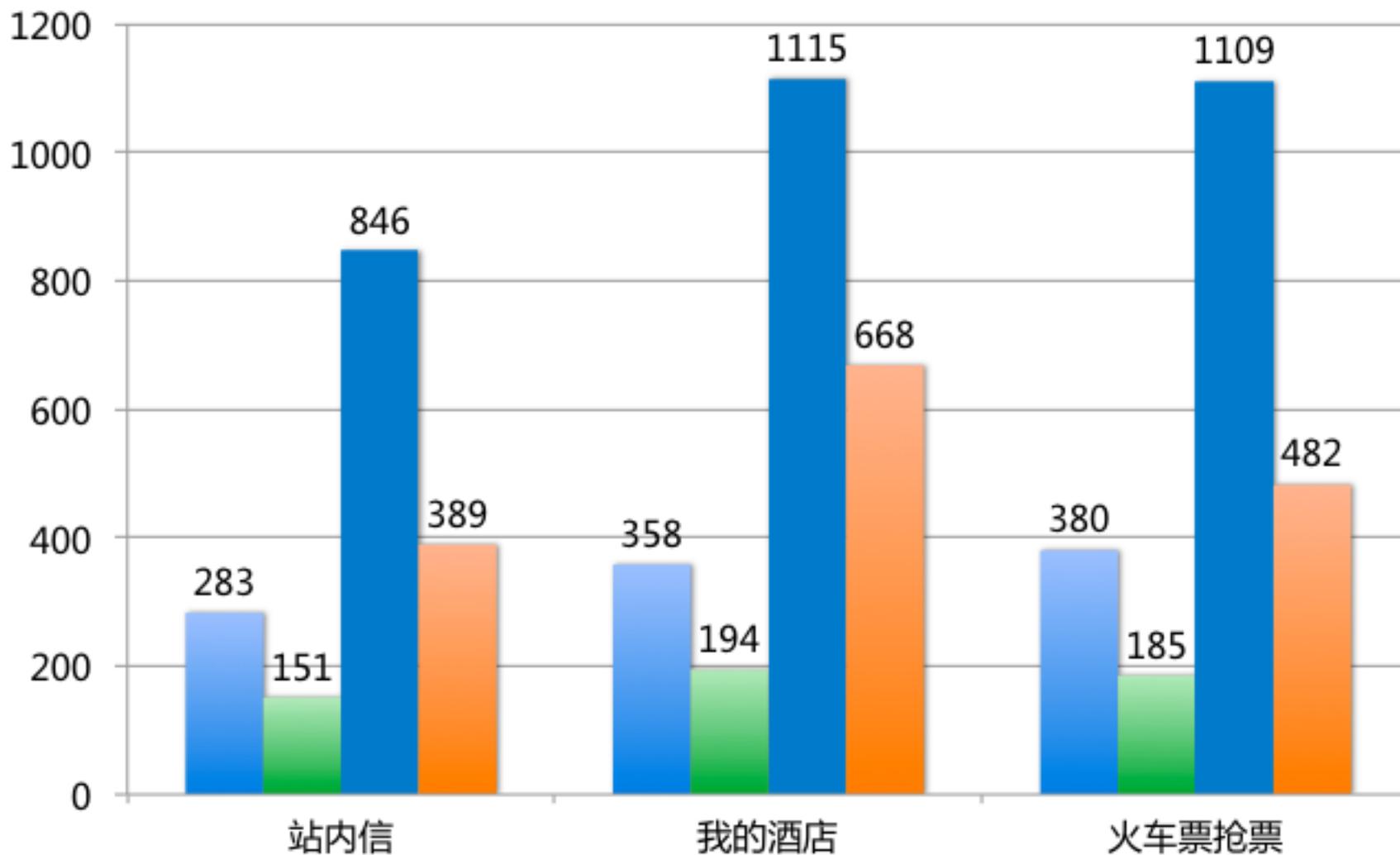
框架Instance管理

- Loading: 框架instance创建过程, 最多2个
- Ready: 框架instance创建完成, 缓存1个
- Dirty: 被业务使用, 缓存1个
- Recycling: 回收过程中, 不缓存
- Error: 出错的instance, 不缓存



注意事项

- Instance内存泄漏风险
- Instance只有初创建的Ready状态可被复用



优化效果

- 整体提升50%
- 实现官方性能瓶颈的优化

- iOS官方加载<ms>
- iOS优化加载<ms>
- Android官方加载<ms>
- Android优化加载<ms>

* 单机多次验证, iOS由iPhone6设备测试/ Android由Sony Speria3测试

携程RN工程实践经验——Develop1



组件设计开发

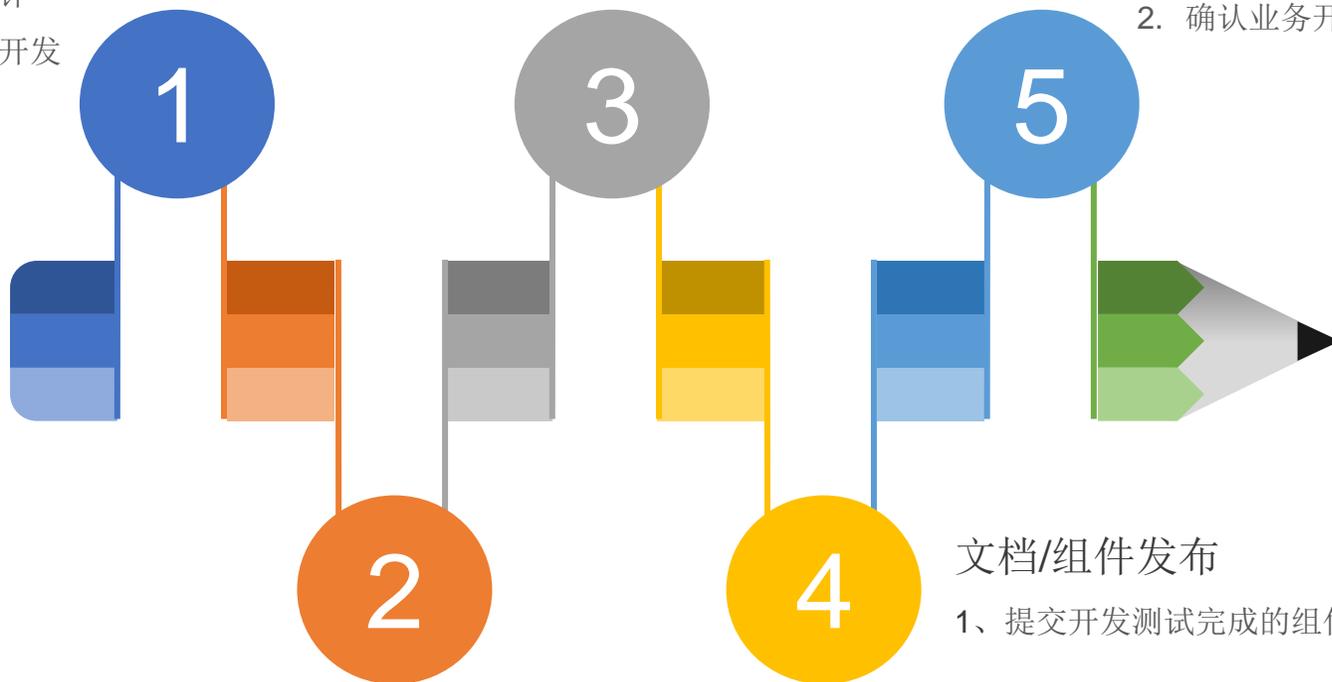
- 1、跨平台设计
- 2、尽量前端开发

Demo验证

- 1、CRNExplorer工程开发Demo

业务使用支持

1. 为业务开发人员提供技术支持
2. 确认业务开发人员的使用



文档/组件发布

- 1、提交开发测试完成的组件到git仓库

使用文档开发生成

- 1、YUIDoc 根据代码注释自动生成
- 2、复杂组件再写文档说明设计和使用

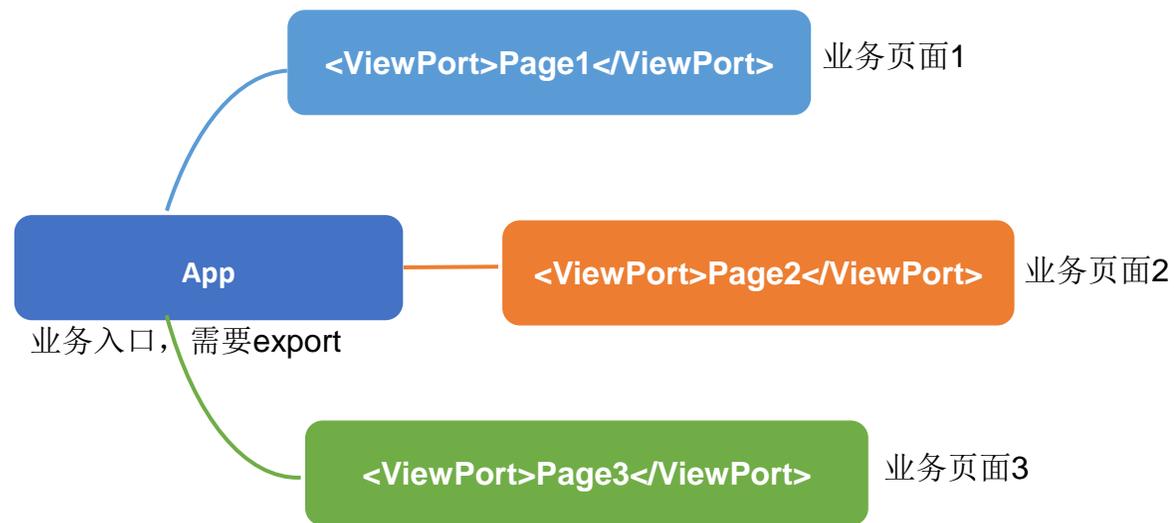
[ABTesting](#)
[AddressBook](#)
[Application](#)
[BirthdayPicker](#)
[Bridge](#)
[Business](#)
[Calendar](#)
[Call](#)
[CRNDev](#)
[Device](#)
[Encrypt](#)
[Env](#)
[Event](#)
[Fetch](#)
[ImagePicker](#)
[Package](#)
[PhotoBrowser](#)
[QRCode](#)
[ScreenShot](#)
[Share](#)
[SOTP](#)
[SOTPCookie](#)
[Storage](#)
[Toast](#)
[UBT](#)
[URL](#)
[User](#)
[Util](#)
[Zip](#)

[App](#)
[Button](#)
[CRNListView](#)
[CRNListViewDataSource](#)
[CustomScrollView](#)
[DatePicker](#)
[DatePickerWidget](#)
[HeaderView](#)
[HtmlText](#)
[LinearGradient](#)
[LoadControl](#)
[LoadingFailedView](#)
[LoadingNoDataView](#)
[LoadingView](#)
[Page](#)
[RefreshControl](#)
[SegmentedControl](#)
[SpriteImage](#)
[SwipeoutView](#)
[ViewPort](#)



App

- AppRegistry.registerComponent注册的模块的基类
- 提供同步、异步的页面路由支持
- 每个业务模块唯一



ViewPort

- 控制全屏页面大小的容器
- 导航栏显示/隐藏自适应大小

Page

- 真实业务逻辑显示的页面
- 需要嵌套在ViewPort中
- 所有业务页面基类



常用控件

- ListView: 内存占用问题
- FlatList: 快速滑动白屏问题
- ScrollView: Header定制问题

CRN解决方案

- 开发CRNListView
- 开发CustomScrollView
- 和原生API基本兼容，降低迁移成本

组件	顶部下拉	底部上拉	右侧索引	建议
ScrollView	支持	支持(需开发)	--	--
CRNScrollView	支持	支持	--	灵活性高，已知item个数，建议使用
ListView	iOS支持、Android不支持自定义UI	支持(需开发)	需开发	--
CRNListView	支持	支持(固定样式)	支持	城市选择类带索引场景
FlatList/ SectionList	固定样式不支持自定义UI	支持(需开发)	需开发	官方在持续优化，建议使用

携程RN工程实践经验——Release&Update1



- ✔ 多版本支持
- ✔ iOS&Android一套业务代码
- ✔ 多RN版本支持(0.30/0.41)
- ✔ 灰度发布、回滚支持

机票退改RN(rn_flight_postservice) [发布历史](#)

Build Filter

[新增版本](#) [配置信息](#) [Mapping配置](#) [Share](#)

版本名称	版本号	版本创建人	FAT	UAT	PRD	发布
	20171011172843		✔发布成功 100%下发中	✔发布成功 100%下发中	✔发布成功 ↻0.00%灰度中	发布
	20171011122747		✔发布成功 100%下发中	未发布 未成功发布	无法发布 未成功发布	发布

携程RN工程实践经验——Release&Update2



• 下载结果

- 99.5%以上的下载成功率
- 支持单用户下载记录查询

• 使用情况

- 统计用户使用最新发布版本的比例

• 包大小优化

- 定制化的bsdiff差分算法
- 差分zip包中的每一项而不是整个文件
- 7z较zip压缩有20-30%的提升空间

• 更新策略优化

- 定时轮训增量包列表
- 失败重试机制

The screenshot shows a release management interface for the package 'rn_flight_postservice'. It includes the following sections:

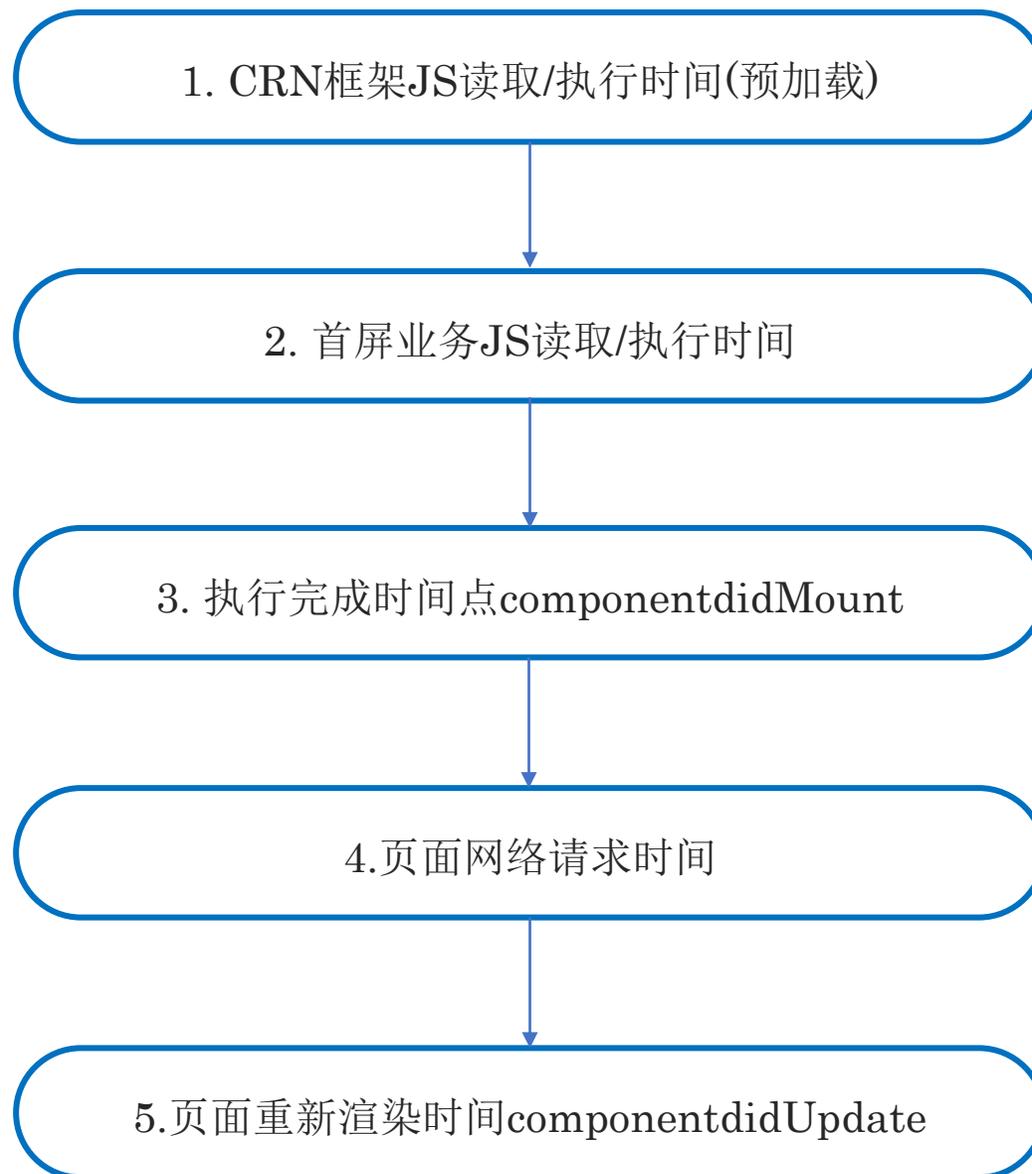
- 发布频道:** rn_flight_postservice
- 目标版本:** 7.8.0, 7.7.2, 7.7.1, 7.7.0, 7.6.3, 7.6.2 (all selected)
- 平台:** Android, IOS, 全平台 (selected)
- 环境:** FAT, UAT, PRD (selected)
- 版本标记:** 普通 (selected), 最低版本, 修改标记
- 下载记录:** 共发布7个包, 共计下载1次 (with a 详情 button)
- 下发包详情:** 下发详情 button
- Status:** 正在灰度
- 操作:** 停止下发, 结束灰度
- 灰度Key:** 20171011191433-ca4741cb-5f38-4a86-a00d-5cdb07dfb05a
- 灰度率:** A progress bar showing 0% (with a slider and markers at 0%, 20%, 40%, 60%, 80%)
- 白名单:** 已配置 8 条Client Id (with a 查看&编辑 button)
- 黑名单:** 已配置 0 条Client Id (with a 查看&编辑 button)

两个概念

- Page基类组件
- CRN容器

性能统计维度

- CRN容器加载性能(无网络请求)
 - 首屏【耗时=1+2+3】
 - 框架层ReactRootView渲染完成点记录
- 业务页面加载性能(含网络请求)
 - 首屏【耗时=1+2+3+4+5】
 - 非首屏(Page切换)【耗时=4+5】
 - 业务埋点记录
- 错误日志上报

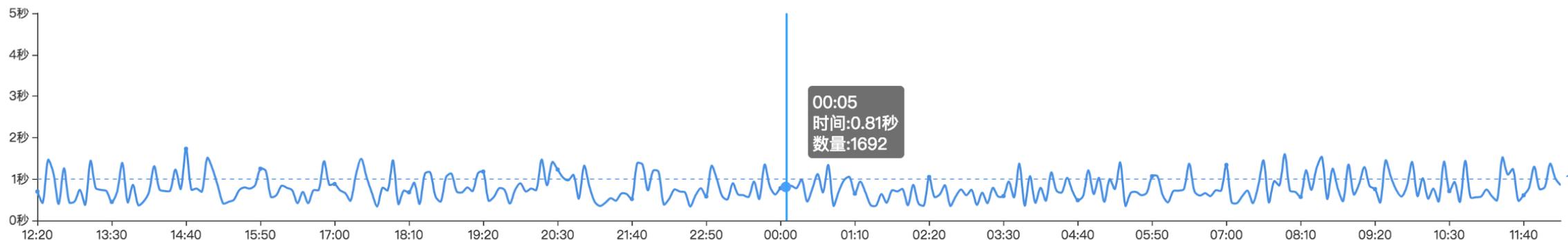


携程RN工程实践经验——Performance2



系统 ALL 版本 7.6.2 时间 最近一天 业务 rn_bus 页面 list 网络 WIFI

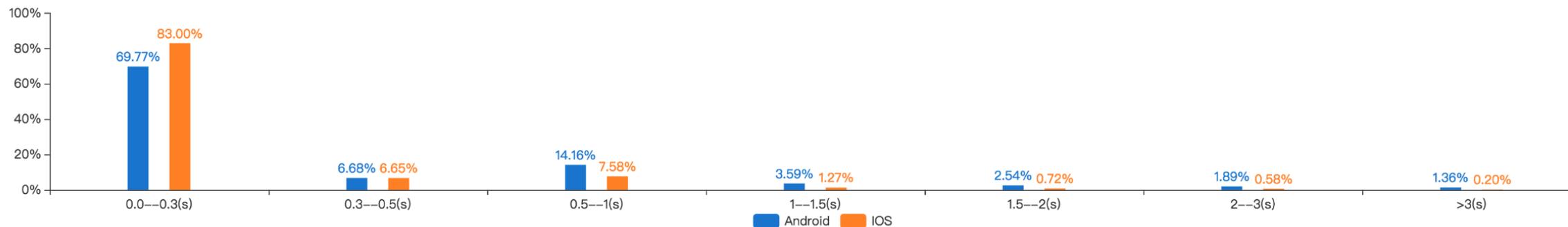
rn_bus业务, list页面加载耗时【业务埋点-包含网络请求时间】(7.6.2版本 - 最近一天), tips



rn_bus业务, CRN容器加载数量和平均耗时【框架埋点-不包含首屏网络请求时间】(7.6.2版本 - 最近一天), tips

#	业务模块	数量	渲染时间(秒)
1	rn_bus	2455303	0.35

rn_bus业务, CRN容器加载耗时分布【框架埋点-不包含首屏网络请求时间】(7.6.2版本 - 最近一天), tips



携程RN工程实践经验——Exception



JavaScript

iOS

Android

JNI Lib

信息Top20	文件Top20	APP版本号	pkgName	pkgIdTop20
is not an object (evaluating 'e.props.passProp... 14.38%	NA 31.84%	7.6.2 100.00%	rn_train_main 49.55%	202586 34.28%
Report-Fatal-Exception: undefined is not an o... 11.65%	app_ctripwebapp_706.000/rn_train_main/js-... 14.05%		rn_bus 41.14%	201278 20.25%
ON Parse error: Unexpected EOF reason: {} 7.97%	webapp_work_7.6.2/rn_train_main/js-module... 11.72%		rn_flight_budg... 3.43%	203000 15.84%
e error: Unexpected EOF 7.89%	app_ctripwebapp_706.000/rn_bus/js-module... 9.96%		rn_common 1.57%	201835 11.28%
defined is not an object (evaluating 'e.ExpVer... 5.64%	app_ctripwebapp_706.000/rn_bus/js-module... 4.49%		NA 1.32%	201671 3.22%
is not an object (evaluating 'e.ExpVersion') 5.55%	app_ctripwebapp_706.000/rn_common/com... 4.14%		rn_hotelorderd... 0.66%	202584 2.61%
is not an object (evaluating 'e.nativeEvent.tex... 4.49%	app_ctripwebapp_706.000/rn_bus/js-module... 3.88%		rn_orderDetail... 0.52%	195292 2.35%
defined is not an object (evaluating 'e.nativeE... 4.46%	webapp_work_7.6.2/rn_train_main/js-module... 3.31%		rn_hotelReserv... 0.42%	unknown_full_... 1.93%
Report-Fatal-Exception: undefined is not an o... 3.10%	webapp_work_7.6.2/rn_train_main/js-module... 3.10%		rn_flight_inquir... 0.28%	NA 1.36%
is not an object (evaluating 's.Platform.OS') 2.82%	app_ctripwebapp_706.000/rn_train_main/js-... 2.70%		rn_message 0.19%	202993 1.08%

1 2 >

时间	信息(message)	文件(file)	pkgName	pkgId	APP版本号 (appVer)	操作系统 (osName)	category	Page ID(pid)
2017-08-23 11:31:16	undefined is not an object (evaluating 's.Platform.OS')	app_ctripwebapp_706.000/rn_train_main/js-modules/20.js	rn_train_main	202586	7.6.2	Android	Crn-error	train_result_rob

用户(vid,sid,pvid)
2452E800697611E782FE102AB3E11BBB, 8, 312

client ID 32001017410018458766	Package 202586	发生时间 2017-08-23 11:31:16	上报时间(服务器端) 2017-08-23 11:31:44
应用版本 7.6.2	设备机型 Redmi Note 3	操作系统名称 Android	操作系统版本 5.0.2

[出错堆栈](#) [原始数据](#)

```
0 <unknown> app_ctripwebapp_706.000/rn_train_main/js-modules/20.js :2 :4023
```

线上CRN错误约0.02%，包括业务JS错误和RN Runtime执行出错

RN运行稳定性方面做了哪些优化？

1. Android端多方位记录错误来源，统一上报错误记录，尤其是JS加载、运行中的相关Fatal Error报错，以及Android JNI层相关处的崩溃的处理；
2. 针对System. loadLibrary失败采用重试机制与上报错误处理，并单独对X86平台的so文件作打包下载，本地加载处理，有效降低java.lang.UnsatisfiedLinkError报错；
3. iOS端对RCTFatal 改造，记录错误来源，对RCTSetFatalHandler 上报错误日志；
4. JS端去除console输出，保证Soft、Fatal的错误信息完整度，区分加载JS前和运行过程中抛出的Fatal Error严重性，分别处理；
5. 建立常见问题的FAQ档案，诸如duplicate props导致运行报错、常用接口undefine或not an object等错误供业务开发查阅；

npm和yarn的使用问题？

1. 采用的git仓库依赖的方式进行package依赖，业务依然使用npm，框架使用yarn依赖；
2. 框架层严格控制使用到的第三方组件版本，打包工具清除业务端重复依赖的package；
3. 控制打包次数，防止出现FAT打包可以，但是UAT或PRD挂了的情况发生；

react-native版本升级问题？

1. 坚持尽量小、尽量少改动源码的原则，采用标记旧版本修改处比对升级到新版本的方式，包括升级打包工具代码和Native/JavaScript库代码；
2. 目前框架层面版本升级大概花费3天左右，每次升级必须回归所有业务代码，诸如接口变更、UI样式、业务逻辑等问题都有可能出现；
3. 打包工具的升级需要着重注意，输入参数、输出结果的改变都需要比对，需要多测试；

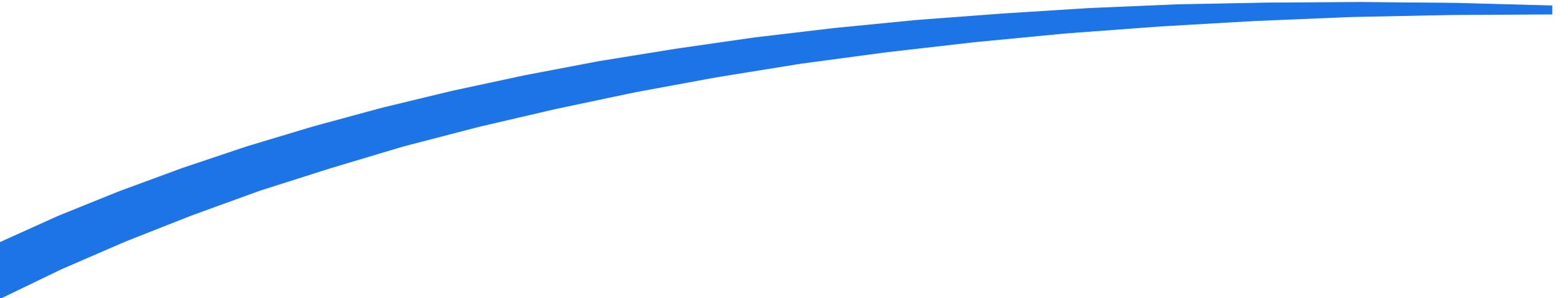
总结与思考

使用RN的几点心得

1. 大量业务使用证明RN的性能和稳定性可靠，适合大规模采用
2. RN开发，能降低业务开发维护成本，满足性能与快速迭代需求
3. 入门难度偏高，要有心理准备，一旦熟悉，可控性强
4. 大规模使用时，最好是有专门团队统筹开发框架和开发发布环境

RN相关待优化点

1. 多业务情况下Instance(Bridge)的可复用性研究
2. 单业务庞大页面数情况下的首屏加载速度优化
3. 官方对FlatList性能表现和稳定性的持续优化，快速滑动白屏问题解决
4. Android端底层库JSCore稳定性提升



THANKS & QA

