OSC源创会 Opensource Innovation Meetup | IT大咖说 知识共享平台

探索H5性能天花板

腾讯NOW直播极限优化实践

腾讯 NOW直播 谢清贵

NOW直播

PROFILE

2014年 携程

旅游事业部

2017年 ~ 至今 腾讯

- 腾讯NOW直播
- IVWEB成员
- 主要负责移动Hybrid APP内业务以及前端性能优化

https://ivweb.io/

APP开发演进

迭代缓慢

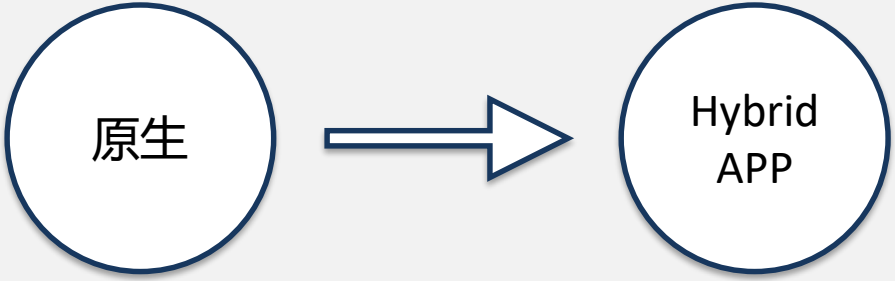
必须跟客户端版本

H5载入速度慢

在线拉取H5耗时很长，用户等待白屏时间长

开发&学习成本较高

安卓&iOS需要两端分别开发一套



客户端安装包很大

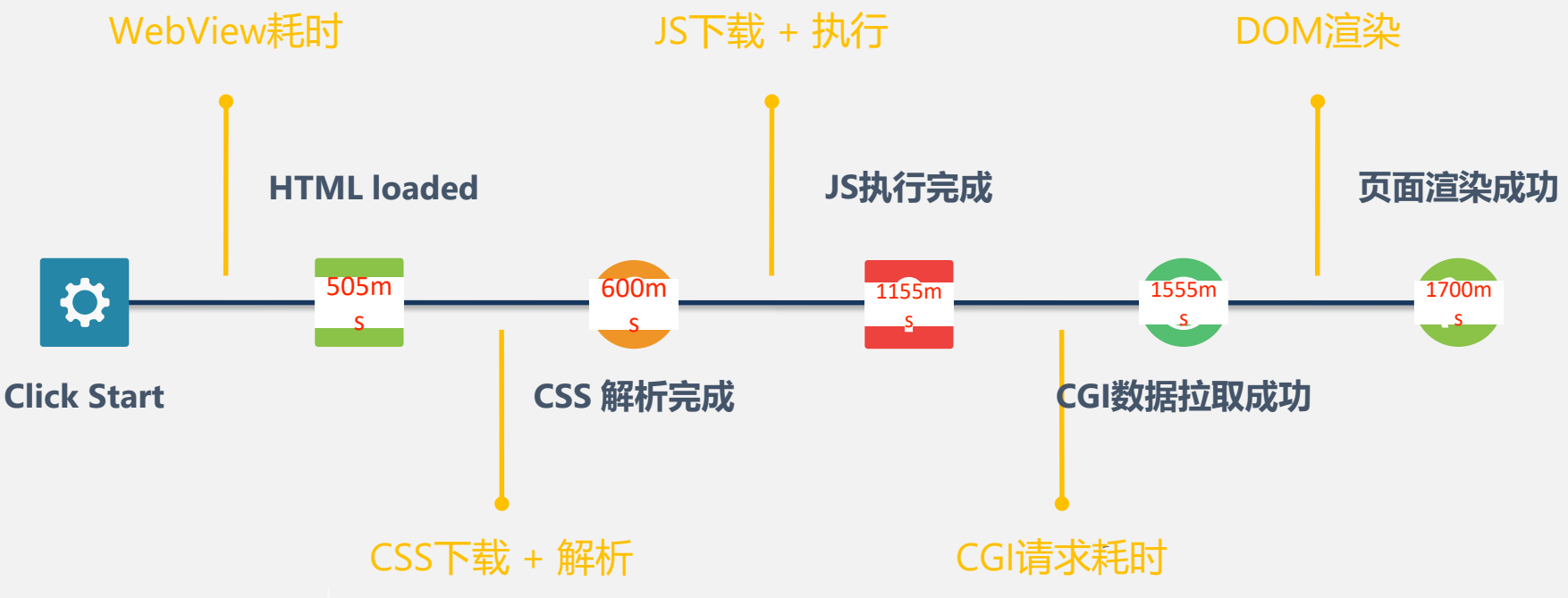
所有业务都必须打包

性能&体验不及原生应用

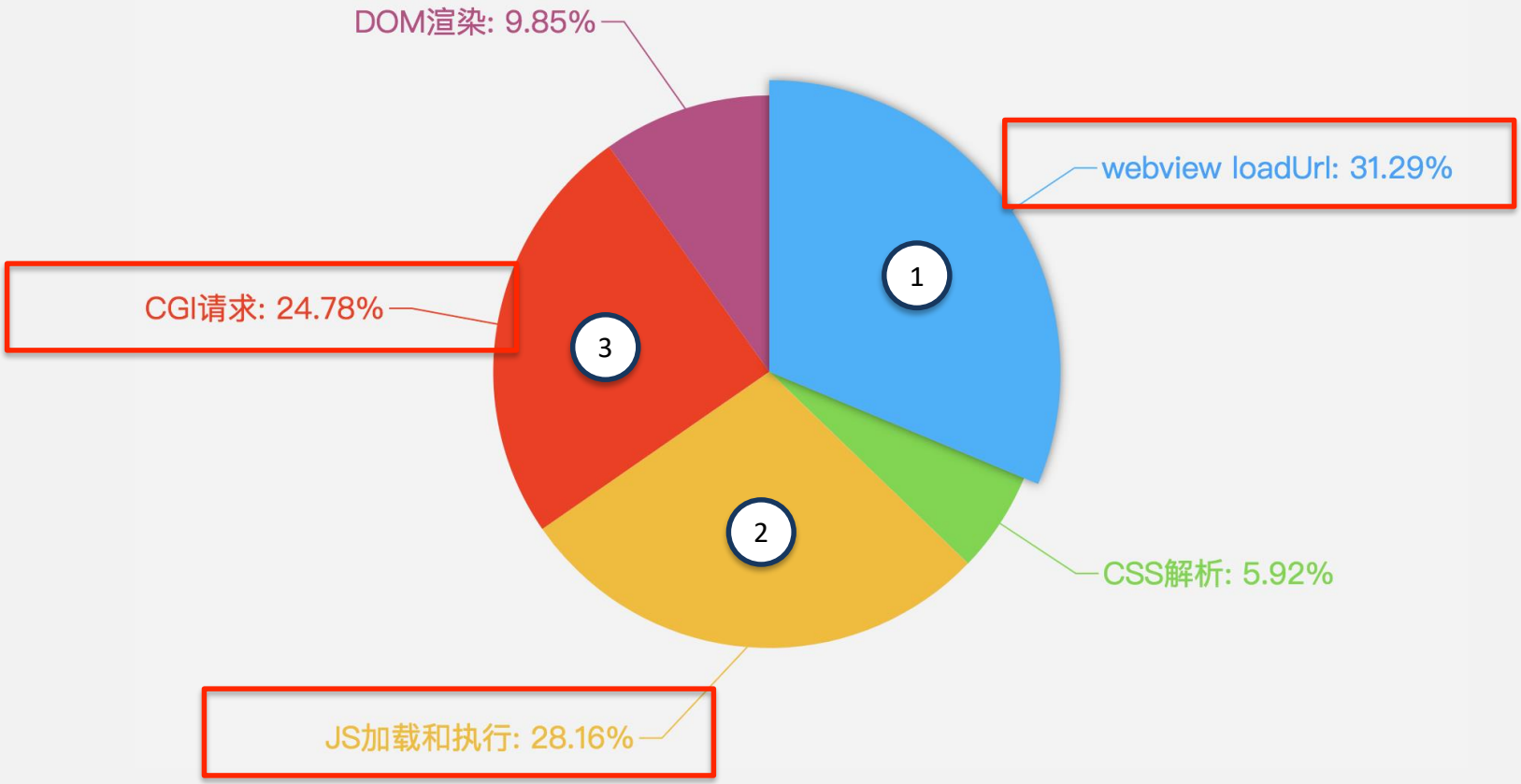
如图片列表无限滚动等


中国移动
22:34
100%
关注
推荐
附近
发现
新年红包天天抢
户外
校园
好歌送上。
92
最后一次
猜猜我在哪
44
抽奖
看不到我
212
131

用户点击到显示首屏的过程



首屏耗时分布



01

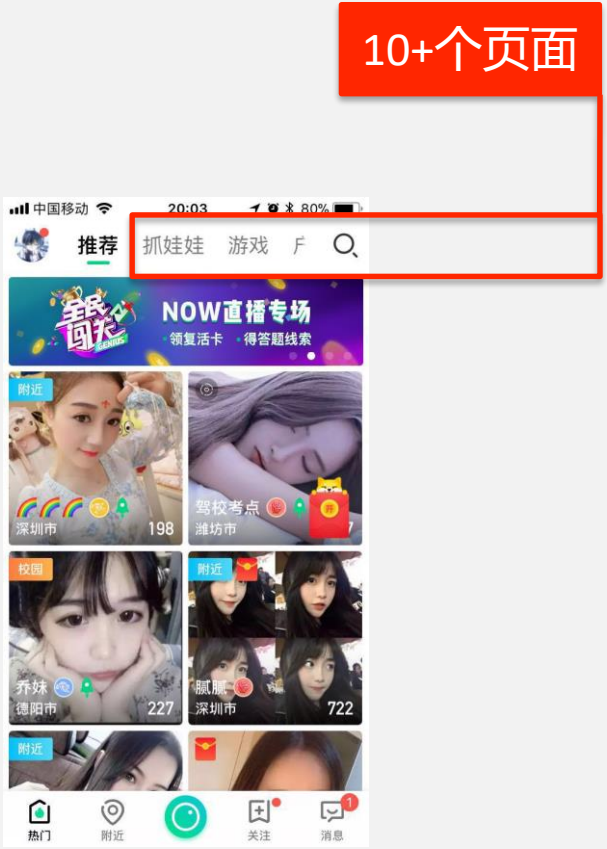
优化WebView耗时

WebView优化方案

页面预加载

打开当前页面的同时，提前预加载其他页面

页面预加载



预加载多少个页面?

- 预加载左右N个
- N = f(cpu, 内存, 机型...)
- f可以是阶段函数

什么时机开始预加载其他页面?

- 当前页面打开时，同时加载所有页面
- 等待当前页面加载成功后，再一起预加载其余页面
- 一个个依次预加载，需要识别加载结束状态

页面预加载— 效果


中国移动
21:38
98%
推荐
抓娃娃
游戏
NOW直播
文明公约
抽奖
新人才艺主播
北京市
292
你的初恋已上线
邯郸市
73
抽奖
嘀嘀嘀，开车了
天津市
290
热门
附近
关注
消息

存在的问题

- 比较浪费用户流量
- 对机器的性能要求很高
- 对未被预加载的页面无效

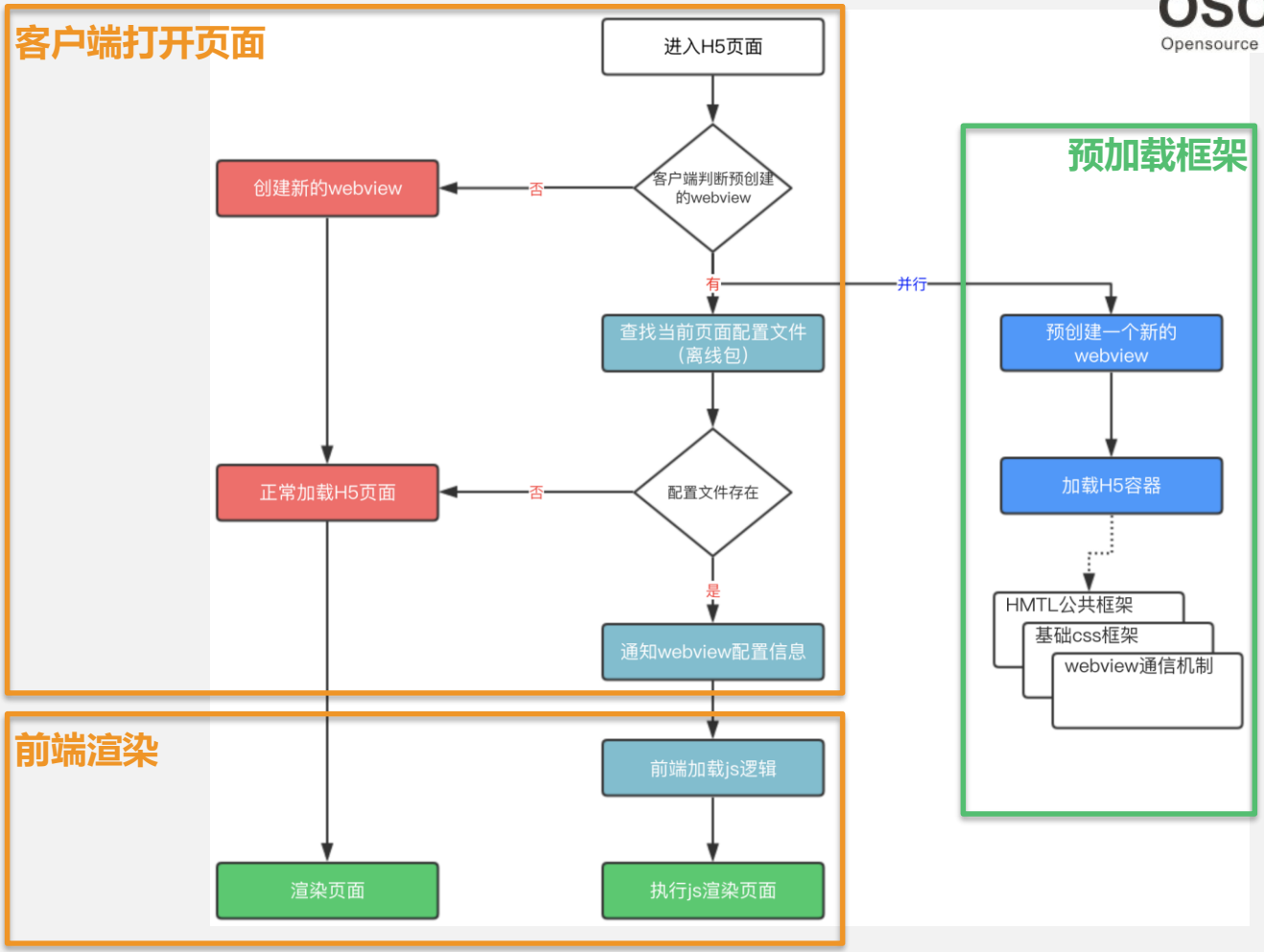
WebView优化方案

页面预加载

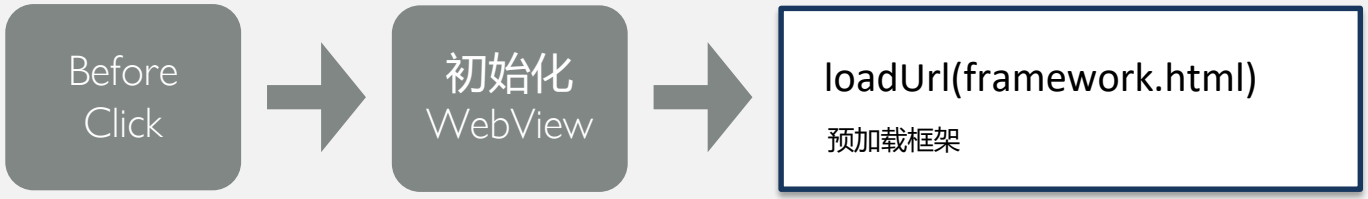
打开当前页面的同时，提前预加载其他页面

WebView H5框架预加载

只预加载框架部分，前端异步加载js并执行业务逻辑



预加载框架



H5页面拆分


framework.html
公共HTML部分（页面结构，公共CSS等）
HTML
H 5
CSS
页面业务CSS
JS
页面业务JS
page

H5页面拆分 — framework.html

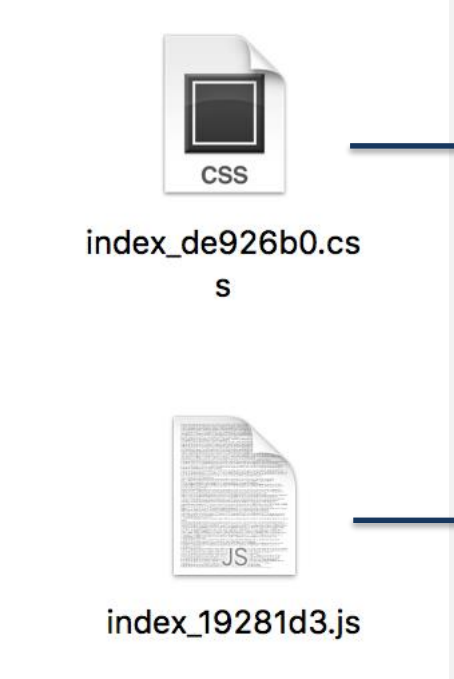

```
<!DOCTYPE html>
<html lang="zh_CN">
<head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=no" />
    ...
    <title>NOW直播</title>
    <script>
        // 记录框架加载时间点
        window._T = {
            framework_page_start: new Date()
        };
    </script>
    <style>
        /* common style */

    </style>
</head>
<body>
    <div id="container"><!--HTML_PLACEHOLDER--></div>
    <script>
        window.__startLoad = function(url, config) {
            // 1. css loader
            ...

            // 2. js loader
            ...
        }

        window._T.framework_page_end = new Date();
    </script>
</body>
</html>
```

1 → 公共CSS

2 → 容器

3 → 入口函数

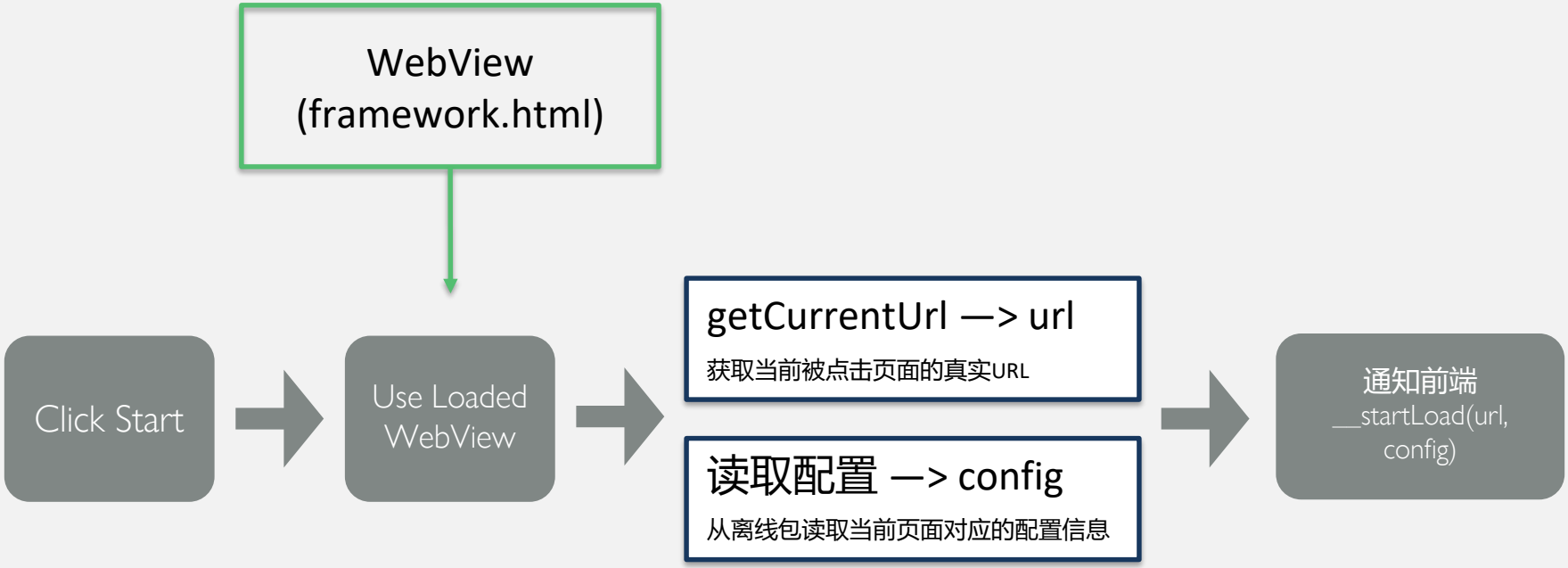
H5页面拆分 — page



业务CSS

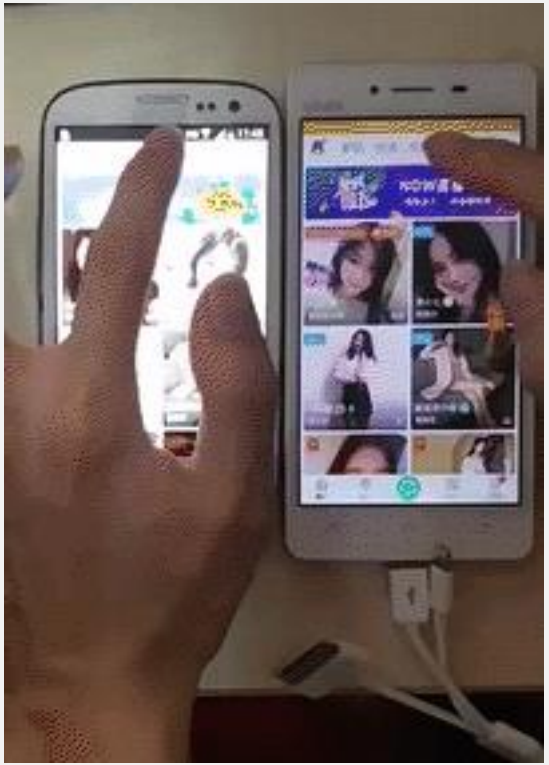
业务JS

客户端打开页面


WebView
(framework.html)
Click Start
Use Loaded
WebView
getCurrentUrl —> url
获取当前被点击页面的真实URL
读取配置 —> config
从离线包读取当前页面对应的配置信息
通知前端
__startLoad(url,
config)

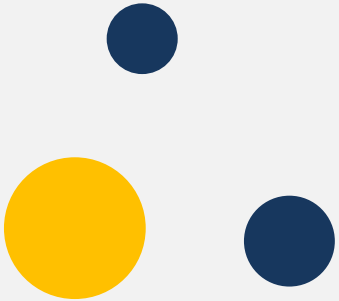
前端渲染

WebView H5框架预加载— 效果



02

优化JS加载&执行耗时



优化JS加载方案

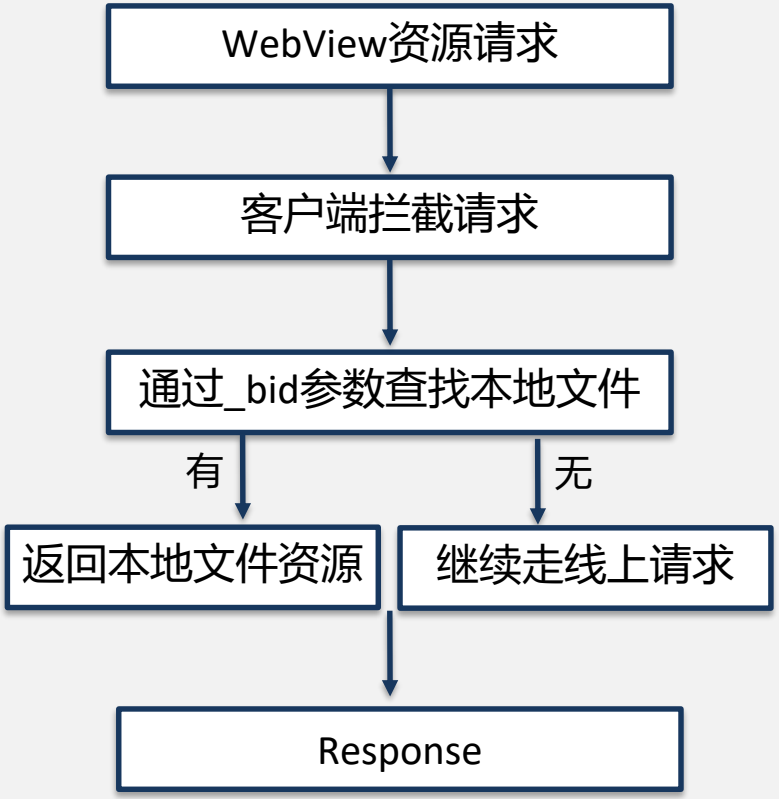
- **PWA**

  利用离线缓存的特性

- **离线包**

  静态资源APP内置

离线包机制



优化JS执行方案

优化框架

升级框架库，剔除无用模块等

优化代码

常用移动端优化手段等

优化效果一般

优化JS执行方案

优化框架

升级框架库，剔除无用模块等

优化代码

常用移动端优化手段等

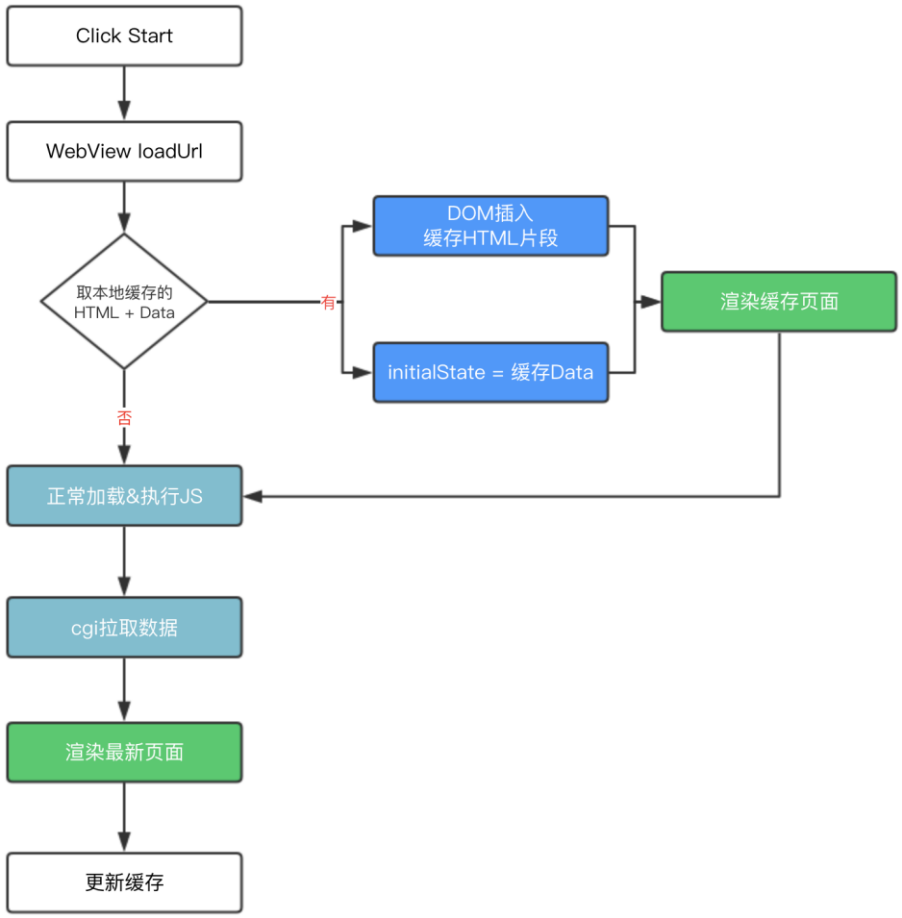
前端本地缓存

将上一次的HTML片段和数据缓存到本地

前端本地缓存

- **前端缓存HMTL + Data**
  缓存首屏的HMTL片段和数据
- **页面打开时，采用本地缓存渲染**
  需要设置redux的initialState，否则会出现一次刷新
- **页面渲染后更新本地缓存**
  在合适的时机点更新

前端本地缓存



缓存机制：

- 客户端提供存储缓存的jsbridge（set）
- 客户端以内存和离线文件的形式存在
- 客户端提供获取缓存的jsbridge（get）

Pros. & Cons.

首屏时间 ≈ WebView loadUrl时间

首屏完全省去JS下载和执行时间 + CGI拉取数据时间，页面秒开

不适用于数据频繁更新的场景

如直播场景等

肉眼可以感知的画面重绘

由缓存HTML到最新数据render会有一次画面刷新

03

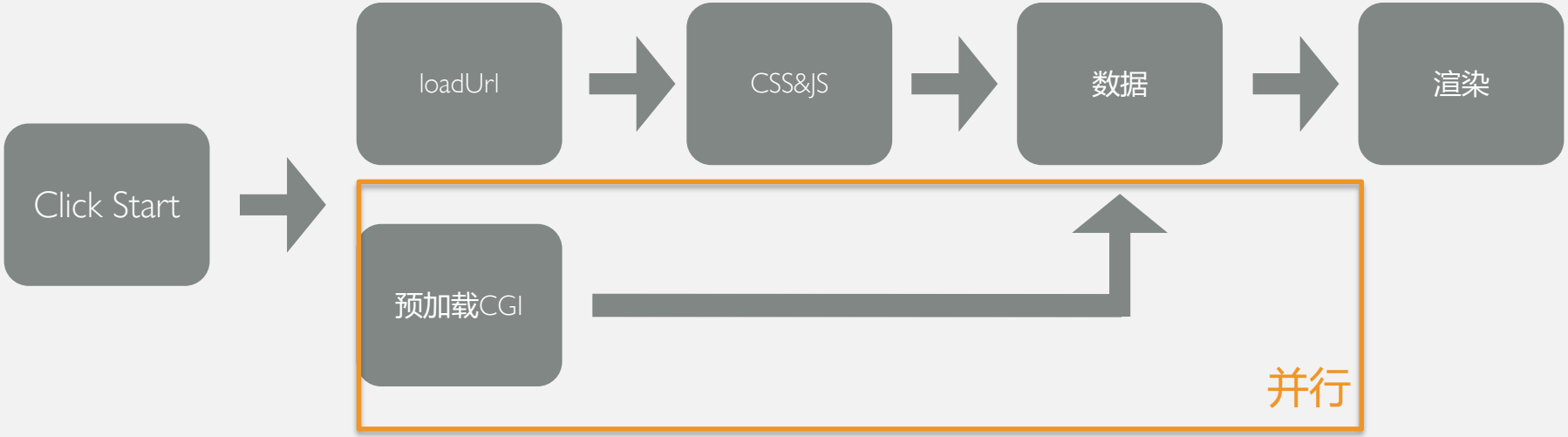
优化数据拉取耗时

优化方案

客户端CGI数据预加载

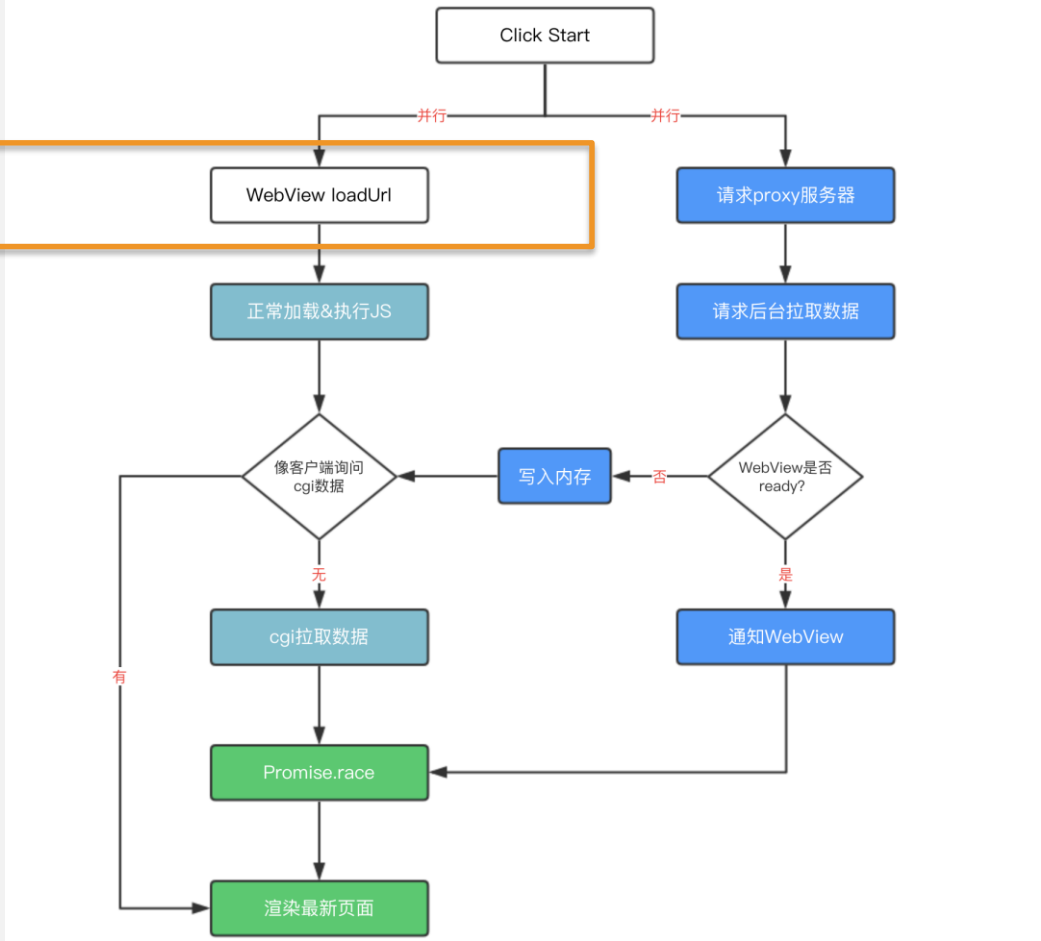
客户端提前帮忙预加载cgi数据

客户端CGI预加载



客户端CGI预加载

充分利用WebView打开等待的这段时间



小结

优化WebView耗时	优化JS加载&执行耗时	优化数据拉取耗时
页面预加载	PWA	客户端CGI预加载
WebView H5框架预加载	离线包	
	前端本地缓存	

THANKS