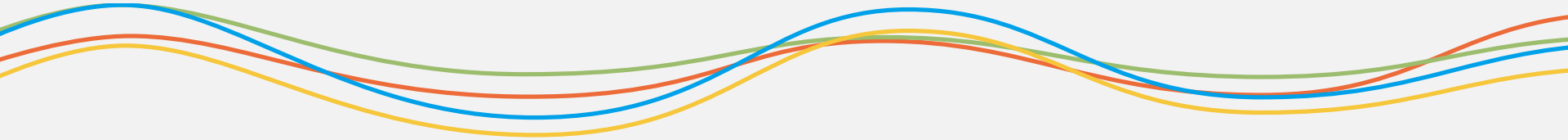
凤凰金融 fengjr.com | 凤凰卫视集团旗下 投资理财平台

听云 TINGYUN | IT大咖说 知识分享平台

ReactNative启动性能优化

张九州



背景

- 为什么跨平台？
  - 效率
  - 动态修复
- 技术选型
  - 原生 + React Native 0.41
- 按业务分bundle
  - 减小bundle体积
  - 局部动态修复
- 双刃剑
  - 加载速度慢
  - 不够稳定

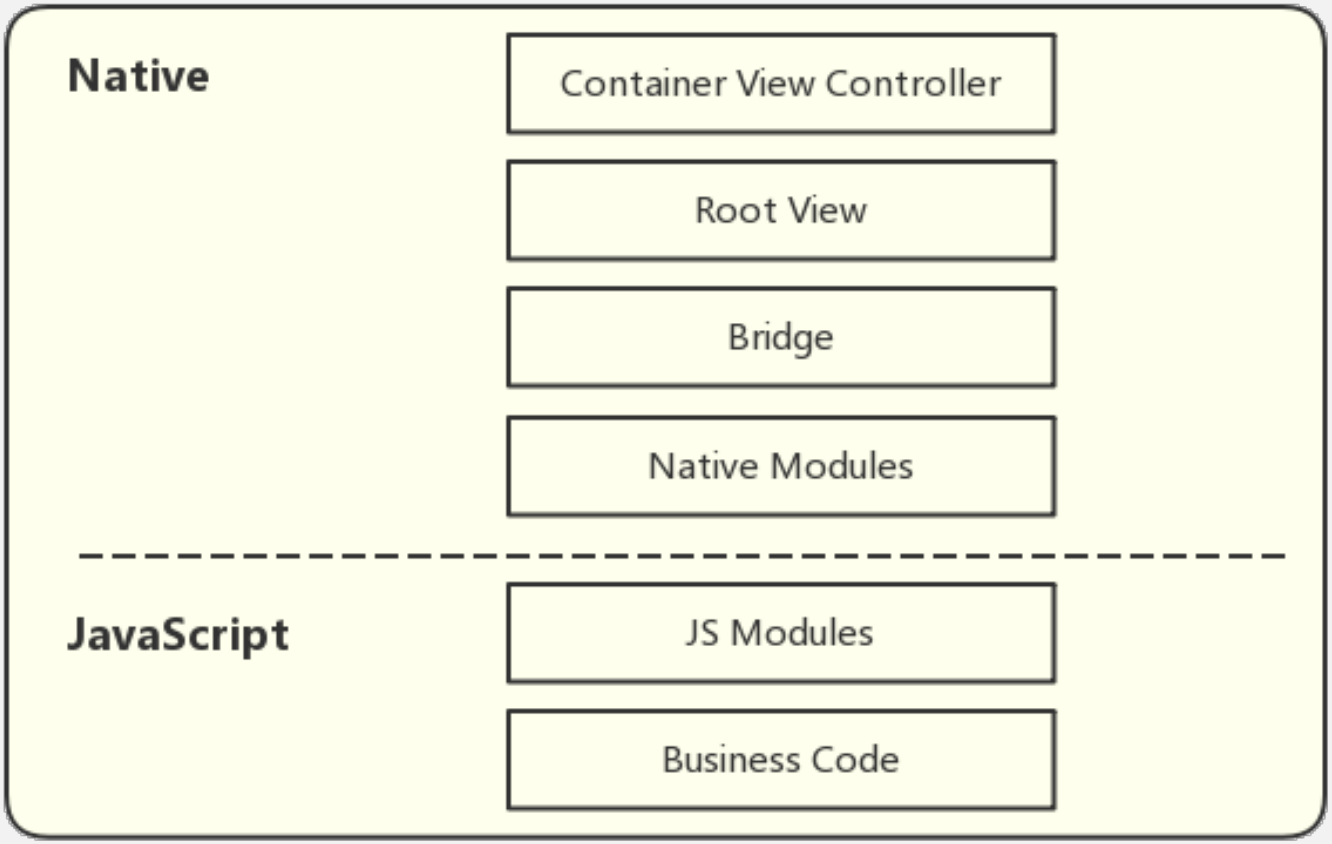
优化之路

- Part1：当前的优化方案
- Part2：未来的优化方向

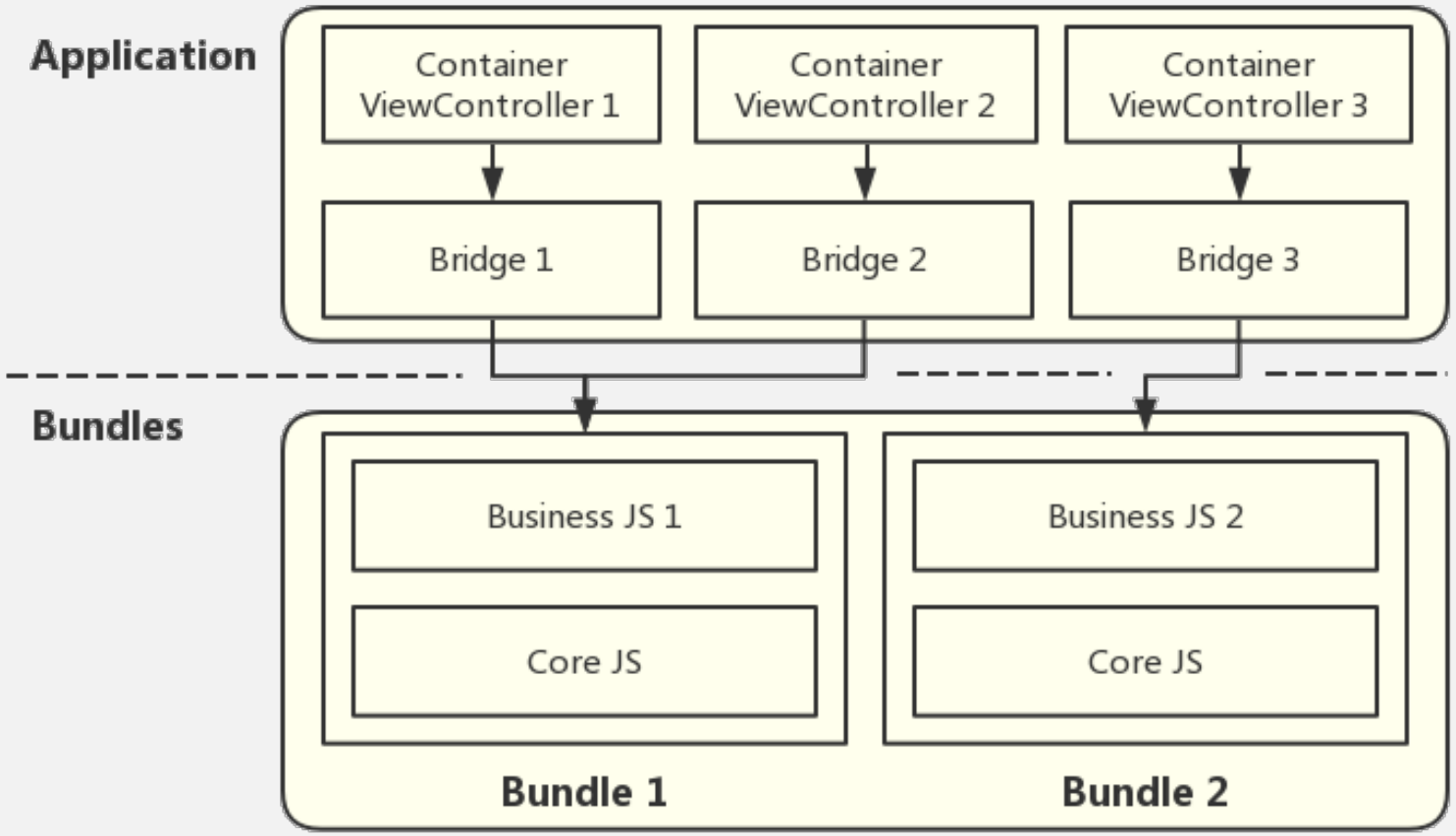
Part1：当前的优化方案

基石

RN页面的组成


Native
Container View Controller
Root View
Bridge
Native Modules
JavaScript
JS Modules
Business Code

第一版架构


Application
Container ViewController 1
Container ViewController 2
Container ViewController 3
Bridge 1
Bridge 2
Bridge 3
Bundles
Business JS 1
Core JS
Business JS 2
Core JS
Bundle 1
Bundle 2

缺陷

- 每次都初始化bridge，慢！

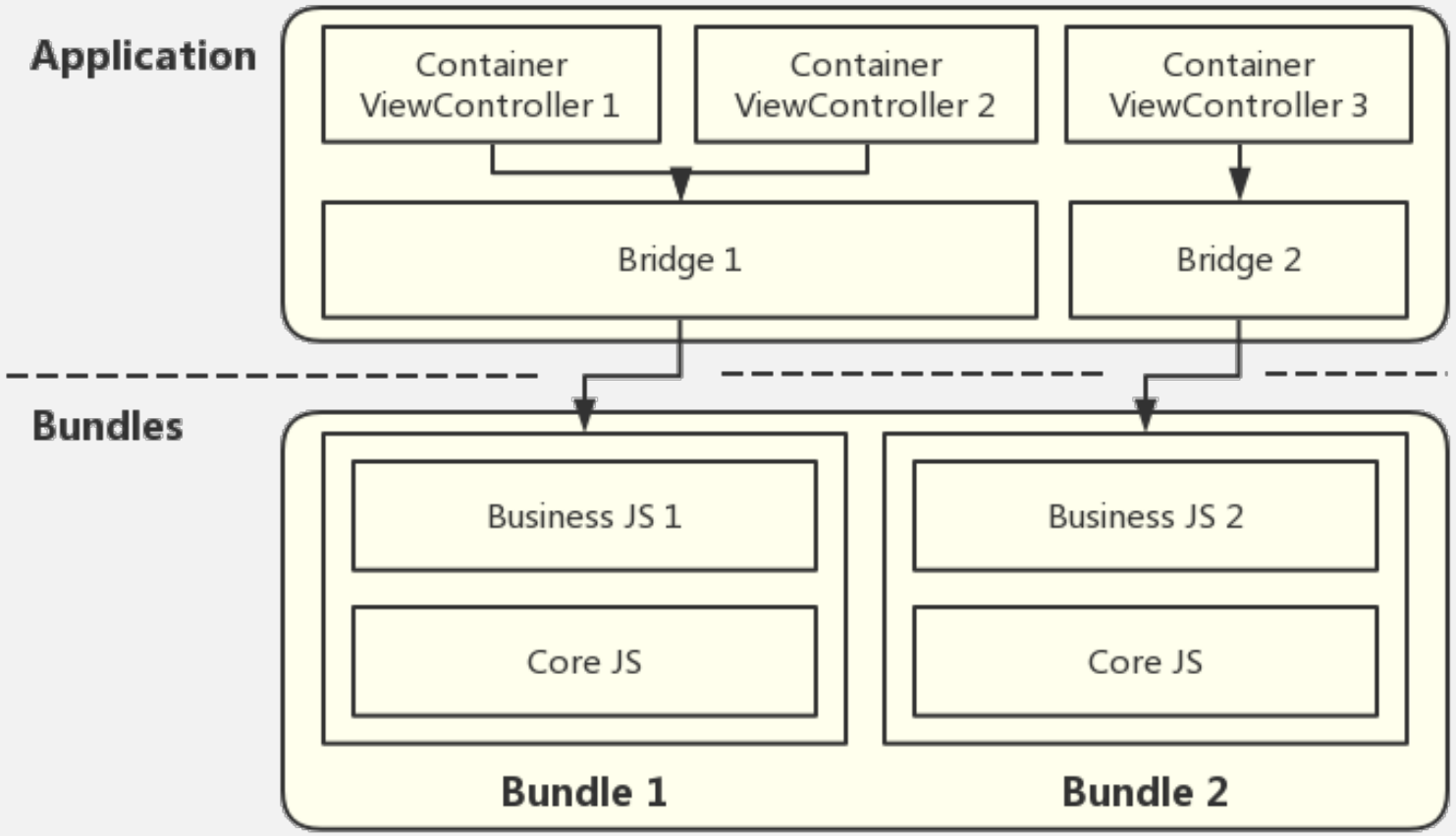
- Bridge实例较多，内存占用高！

优化方案

- Bridge全局缓存，只初始化一次。
    - 提高二次加载速度

- 同业务共享bridge。如果有N个业务，最多有N个bridge实例。
    - 提高同业务页面加载速度，减少内存使用。

第二版架构


Application
Container ViewController 1
Container ViewController 2
Container ViewController 3
Bridge 1
Bridge 2
Bundles
Business JS 1
Core JS
Business JS 2
Core JS
Bundle 1
Bundle 2

实现

- Bridge复用接口
  - 集成复用和缓存逻辑
- 副作用
  - 避免冲突
  - 并发加载
  - 留意“初始化”代码

Bridge复用接口

```
RNBundle *bundle = [[RNBundleManager sharendInstance]
bundleWithName:name];
[bundle loadBridge:^ (RCTBridge *bridge) {
      RCTRootView *rootView = [[RCTRootView alloc]
initWithBridge:bridge
                                              moduleName:@"mobile"
                                              initialProperties:[mProps copy]];
       rootView.frame = self.view.bounds;
      [self.view addSubview: rootView];
}];
```

避免冲突

一个Bridge对应多个界面，可能造成数据冲突！

RN为每个root view分配唯一root tag，并同步到JS。可利用此标识来区分不同root view的数据。

以下类型数据可能发生冲突：

- 全局变量
- 单例
- 事件

并发加载

用户有可能快速切换页面，导致重复加载。

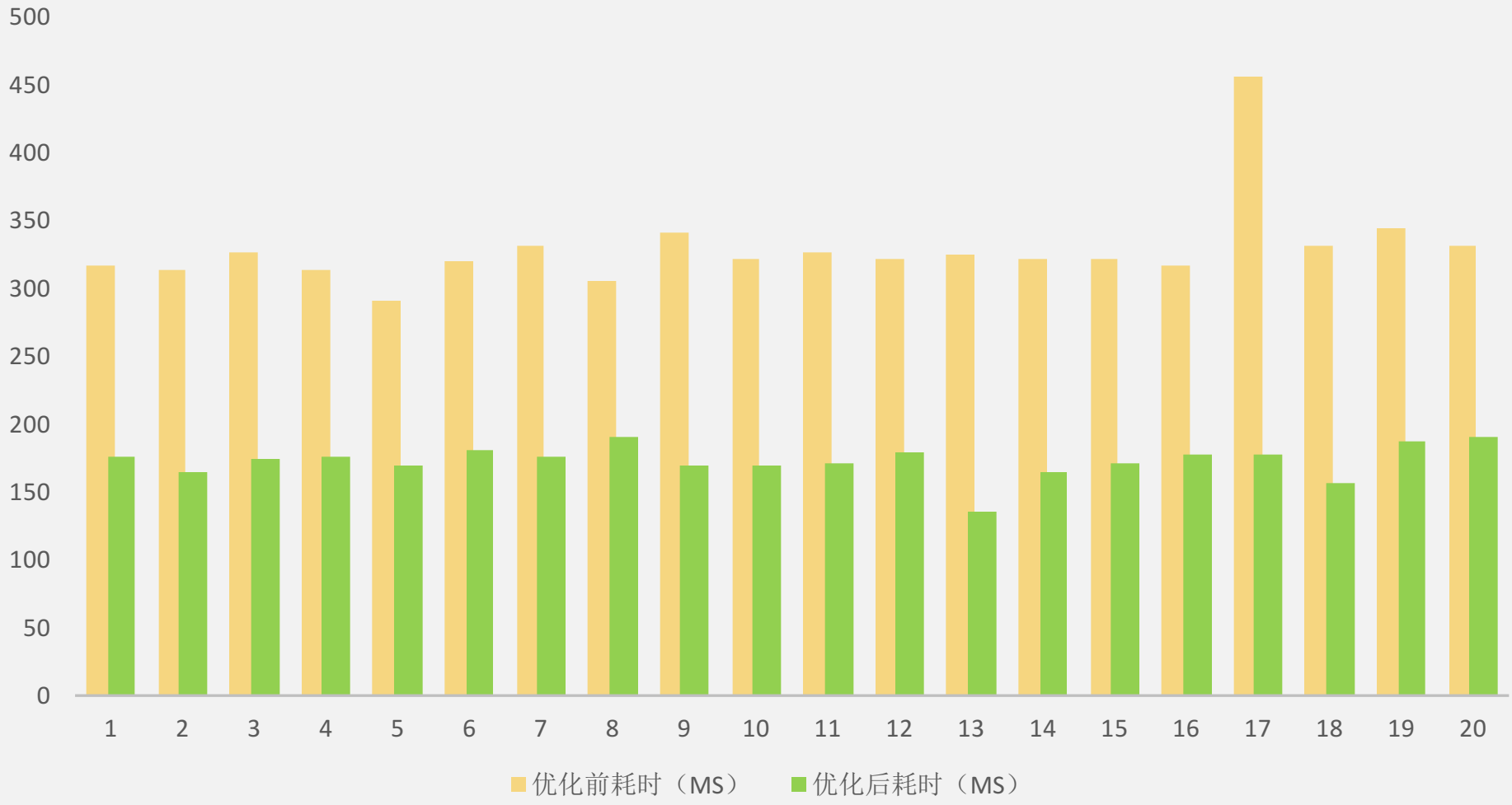
- 标记某个bridge是否正在加载。
- 如果在加载中，则不执行加载。
- 在加载完成后，统一调用回调block。

留意“初始化”代码

听云 TINGYUN | IT大咖说 知识分享平台

Bridge复用可能导致一些只需要执行一次的代码被执行多次。

性能测试（单个页面第二次加载）



总结

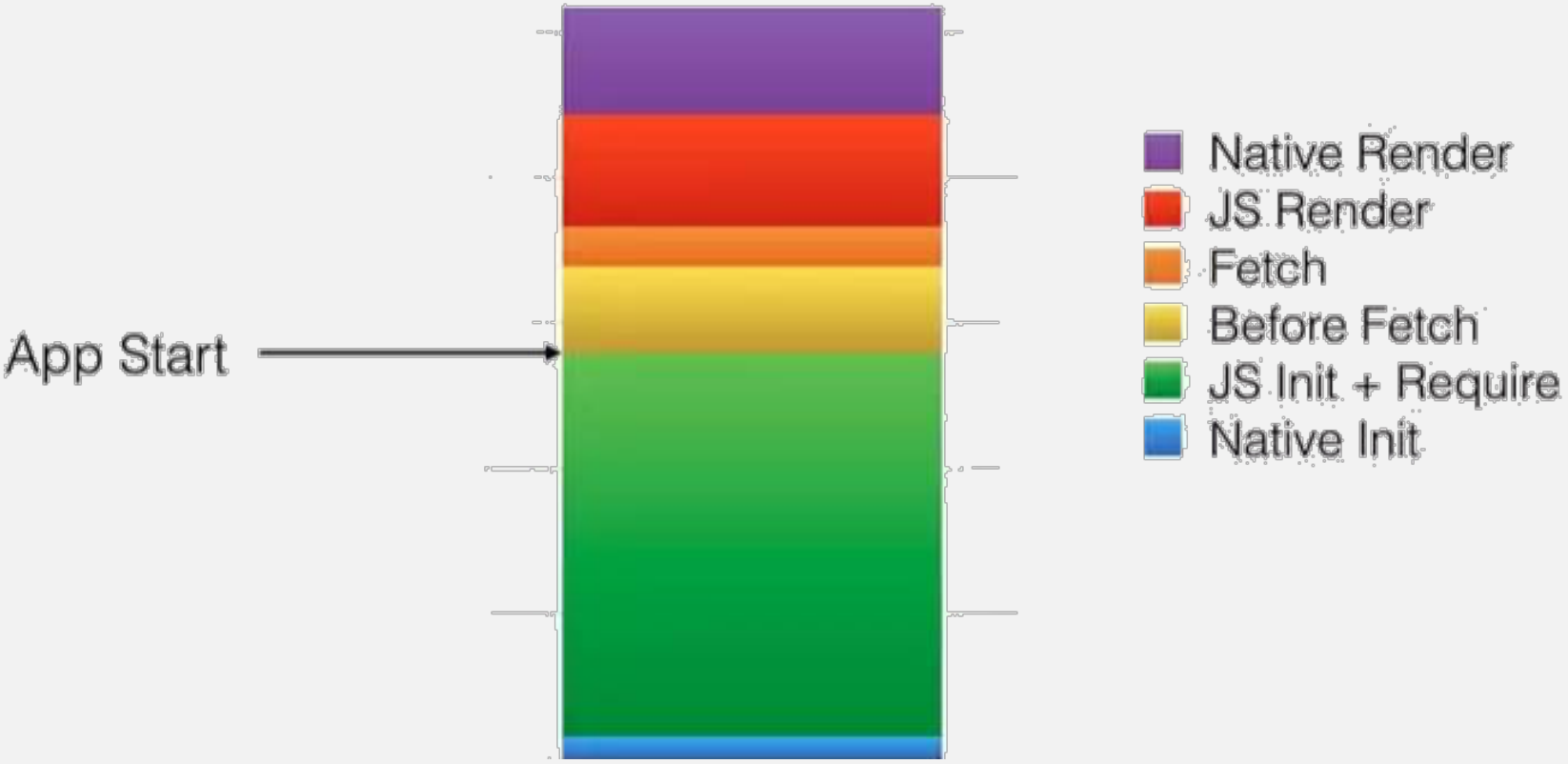
- 优势
  - 简单易实现。
  - 对RN源代码无侵入。
- 劣势
  - 没有做到核心、业务代码拆分，造成核心代码多次加载。
  - 核心部分JS不能被多个业务共享。
  - 不同业务之间不能通过JS来跳转，只能推新的Container VC。

还是不够快！

Part2：未来的优化方向

探究

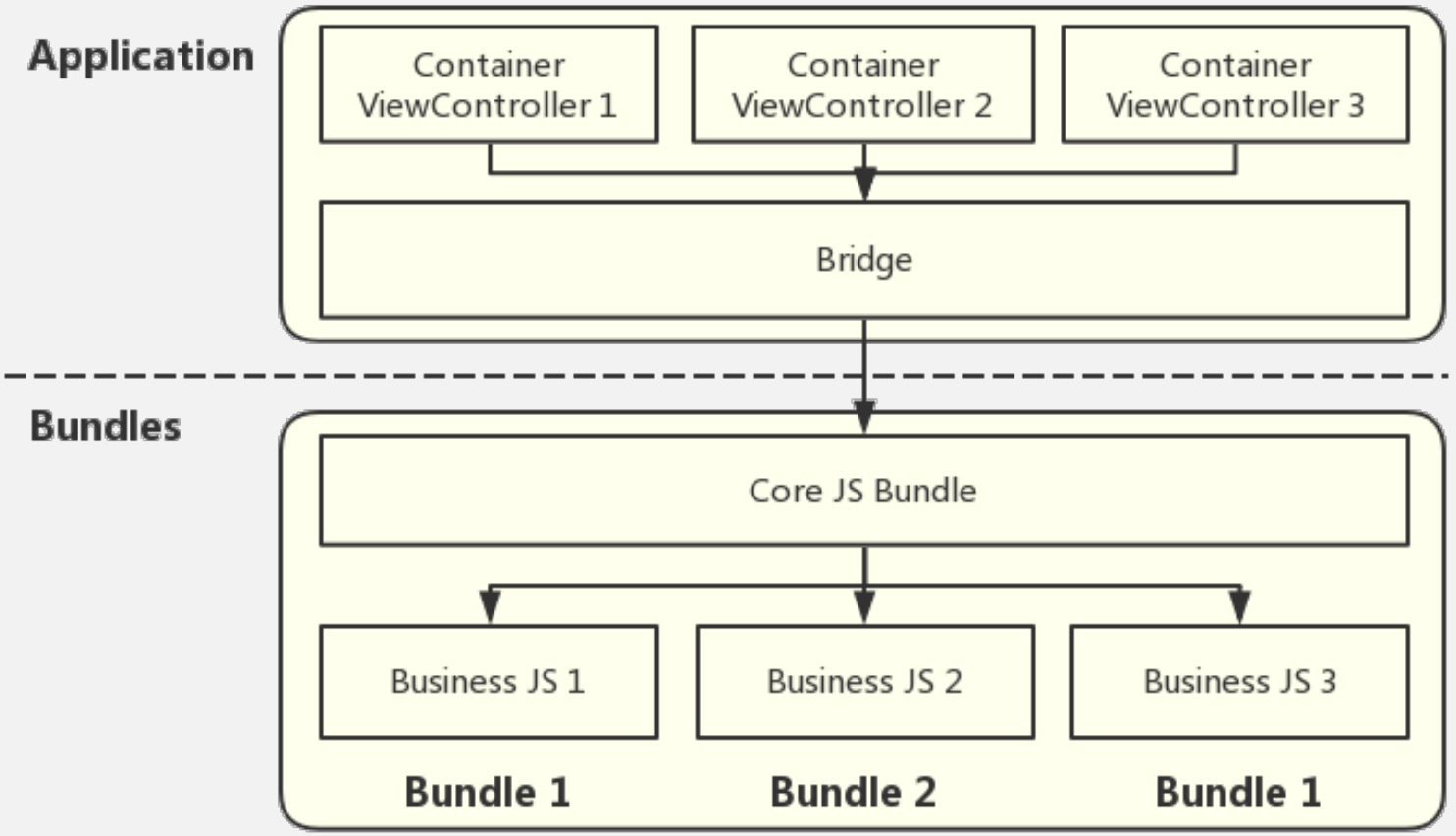
RN页面加载的时间消耗



我们希望

- 只有一个bridge。
- 核心代码和业务代码分离到不同bundle。
- 核心bundle只加载一次。
- 业务bundle实现懒加载。

听云 TINGYUN | IT大咖说 知识分享平台



向着目标前进

- 探究bundle文件结构。
- 修改打包流程。
- 实现业务包懒加载。

探究bundle文件结构

听云 TINGYUN | IT大咖说 知识分享平台

- 三个组成部分：
  - 定义polyfills。
  - 定义模块（JS文件）。
  - 执行入口代码。

- ModuleId
  - Bundle中JS模块的唯一标识。
  - 定义、获取模块都通过这个标识。

修改打包流程

- 生成全局唯一moduleId。
  - 使用文件存取用过的moduleId。
  - 下次打包时，直接取用上次生成好的moduleId。如果没有，产生一个新id并存储。

- 分离核心、业务代码。
  - 使用模块路径判断是否为核心代码。根据条件过滤掉不需要的代码。
  - 业务包在最后手动require入口模块。

生成全局唯一moduleId——代码示例

```
// 生成moduleId
function createModuleIdFactory() {
  return ({path: modulePath}) => {
    if (!(modulePath in fileToIdMap)) {
      global['fileToIdMap'][modulePath] = global['fileToIdMap']['nextId'];
      global['fileToIdMap']['nextId'] += 1;
    }
    return global['fileToIdMap'][modulePath];
  };
}


// 读文件
var fs = require('fs')
var module2Id = fs.readFileSync('/Users/admin/module2Id', 'utf-8')
global['fileToIdMap'] = JSON.parse(module2Id) || { nextId: 0 }

// 写文件
fs.writeFile(‘/Users/admin/module2Id’, JSON.stringify(global['fileToIdMap']), err => {
  console.log(err)
})
```

实现业务包懒加载

听云 TINGYUN | IT大咖说 知识分享平台

- 核心包包含RN源码、路由系统。
- 业务包包含本业务所有页面的路由配置、所有业务模块。
- 跳转到一个未注册路由的页面时，尝试加载业务包。需要维护页面和业务包的关系。

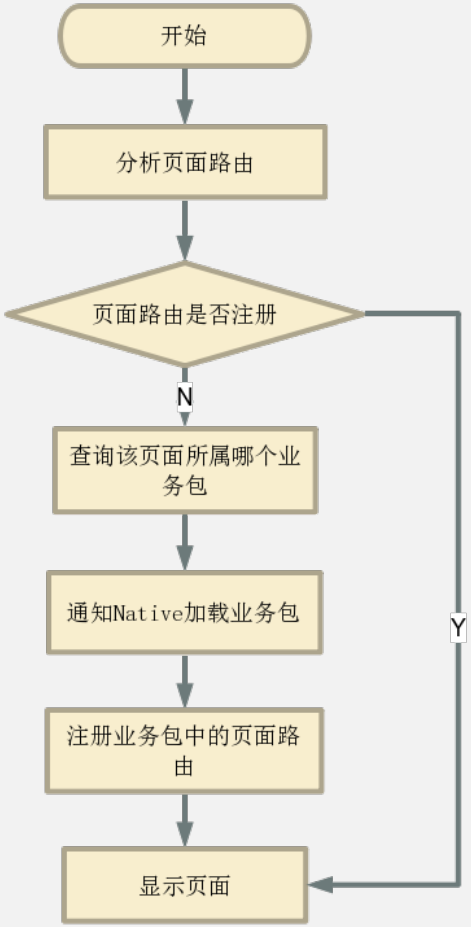
懒加载 – JS代码示例

```
function loadPage (navigator, name) {
  if (!route.hasRoute(name)) {
    let map = {
      'page1': 'business1',
      'page2': 'business1',
      'page3': 'business2'
    }

    NativeModules.RNDynamic.loadJSBundle(map[name]).finally(() => {
      navigator.push(route.getRoute(name))
    })
    return
  }

  navigator.push(route.getRoute(name))
}
```

懒加载 – Native代码示例

```
RCT_EXPORT_METHOD(loadJSBundle:(NSString *)bundleName fullfil:(RCTPromiseResolveBlock)fullfil reject:(RCTPromiseRejectBlock)reject) {
  NSString *bundleURL = [[NSBundle mainBundle] pathForResource:bundleName ofType:@"jsbundle"];
  if (!bundleURL) {
    reject(@"0", @"0", nil);
    return;
  }
  NSData *jsData = [NSData dataWithContentsOfFile:bundleURL];

  id<RCTJavaScriptExecutor> exe = [self.bridge moduleForName:RCTBridgeModuleNameForClass(NSClassFromString(@"RCTJSCExecutor"))];
  [exe executeApplicationScript:jsData
                      sourceURL:[NSURL URLWithString:bundleURL]
                     onComplete:^(NSError *error) {
                       if (!error) {
                         fullfil([NSNull null]);
                       } else {
                         reject(@"0", @"0", error);
                       }
                     }];
}
```

懒加载流程


开始
分析页面路由
页面路由是否注册
N
Y
查询该页面所属哪个业务包
通知Native加载业务包
注册业务包中的页面路由
显示页面

总结

听云 TINGYUN | IT大咖说 知识分享平台

- 建立在第一阶段优化的基础上。
- 需要修改打包脚本，相对复杂。
- 单bridge，核心业务分离，本质上提高性能。

THANK YOU