hybrid开发分享

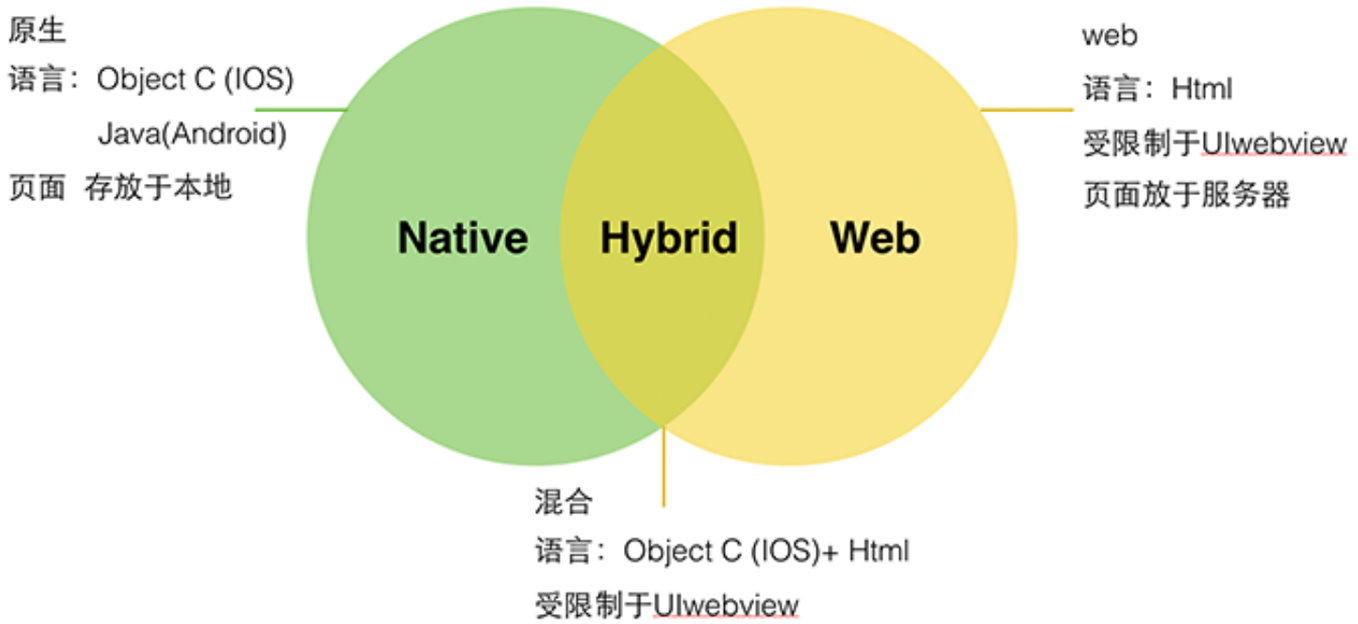
— 邵乾飞

目录

- 什么是hybrid app?
- hybrid的优缺点
- 怎么做hybrid?
- js和app的交互方案
- 项目实战中的技术点和问题解决
- 更优雅的实现方式?

什么是hybrid app?

什么是hybrid?



- hybrid app的命名，主要是为了区别于native app和web app, 兼具“Native App良好用户交互体验的优势”和“Web App跨平台开发的优势”

- 是一种用户交互界面尽可能采用html, 底层交互(如调用相机，文件系统，重力感应等)采用原生程序的混合模式

hybrid的优缺点

hybrid的优缺点

优点:

- html+css+js的跨平台特性, 大大减少开发量
- css排版速度快于native UI开发,比较适合复杂排版内容呈现
- js+css等静态资源文件独立于app管理，可以快速更新而不需要发包、审核等流程

缺点:

- js执行效率不如原生代码
- 浏览器渲染效率不如native
- 浏览器本身的不足: 弹性效果、触摸、滚动不如人意

怎么做hybrid?

几种技术选型的优缺点

技术选型	优势	劣势
IONIC	• 文档多、技术成熟 • ios 和 android 基本上可以共用代码 • 实质是web开发，开发起来比较容易	• 占用内存高，不适合做游戏类型app • 耗性能的地方无法利用native的思维实现优势互补，如高体验的交互，动画
React-Native	• 开发效率较高 • flexbox 布局，更加简单高效 • ios 和 android 基本上可以共用代码	• 学习要求高，必要时需要懂一些native的东西去扩展
WEEX	• 核心语言是vue，上手容易 • 能支持Ios、Android、web	• 社区不如RN,生态不好 • 坑比较多
Flutter	• 无需桥接，原生编码 • 能支持Ios、Android (更多?)	• 学习新语言Dart • 框架重
Native	• 最好的体验以及功能实现 • 成熟、完善	• 学习要求高 • 开发成本高、周期长

怎么做hybrid?

利用第三方集成开发工具:

- framework7
- phone gap
- IONIC
- Cordova
- AppCan

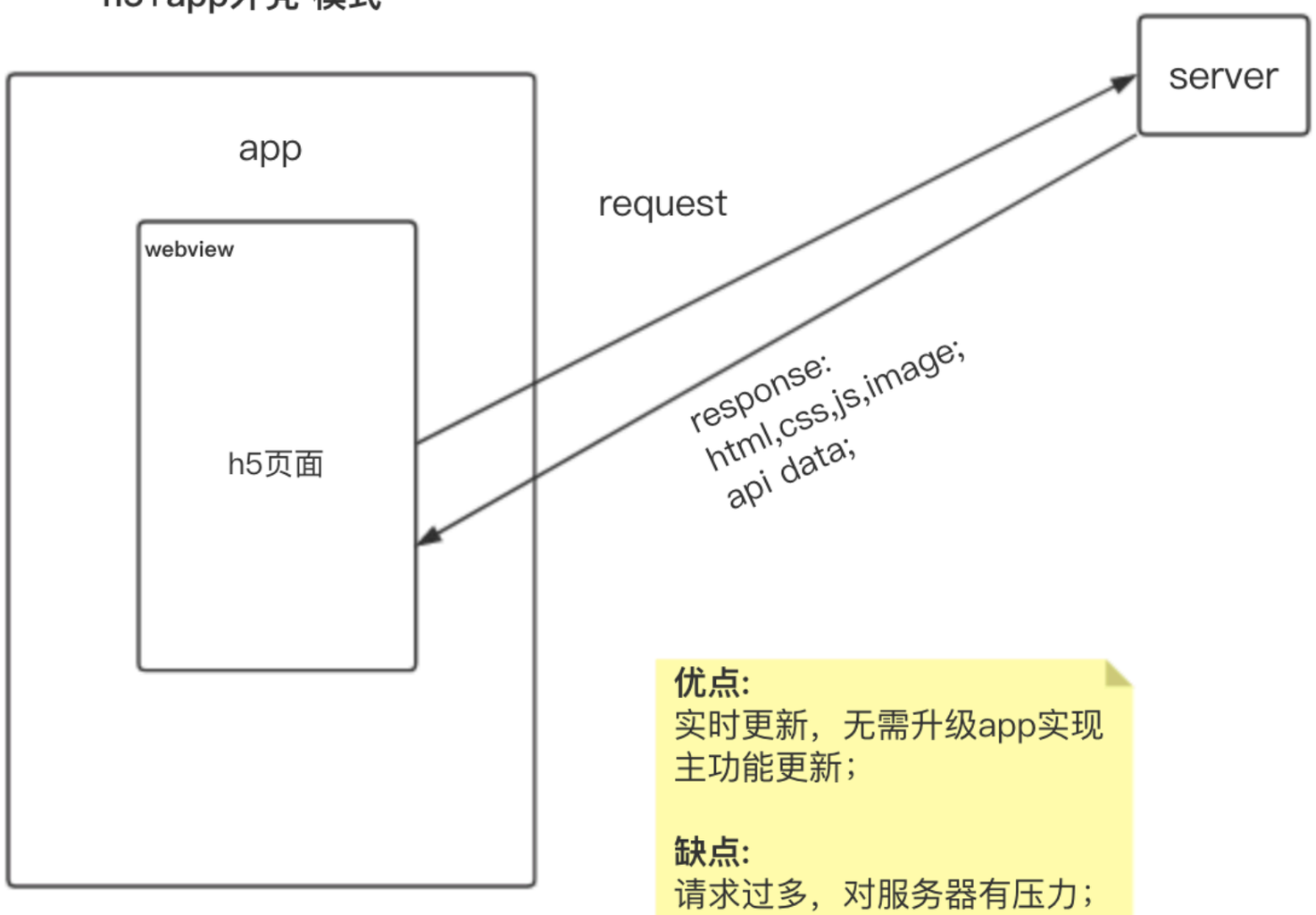
分离式开发:

js+css+html 开发完毕，然后放入app的内嵌webview中， 需要app端配合

(提供一个供开发调试的apk/ipa包安装在真机上)

分离式开发的几种模式

h5+app外壳 模式

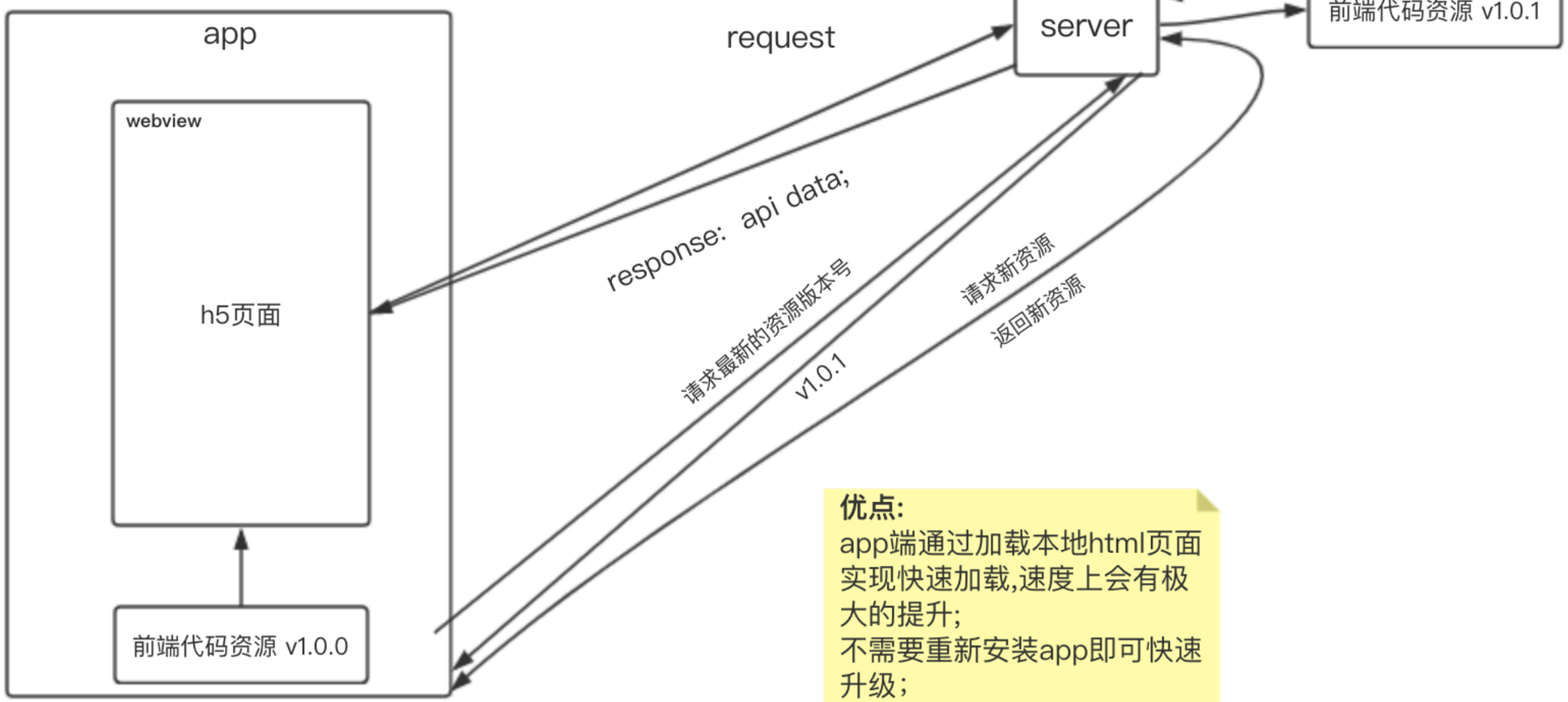


优点:
实时更新，无需升级app实现主功能更新；

缺点:
请求过多，对服务器有压力；
页面加载慢；

分离式开发的几种模式

离线web页面 模式



原理:
需在服务器上建立对应的资源包，并将服务器资源包与本地html版本做对比，若不一致则从服务器加载资源压缩包，并下载到本地

优点:
app端通过加载本地html页面实现快速加载,速度上会有极大的提升;
不需要重新安装app即可快速升级;

缺点:
需要技术支持，要求会比在线web页面方案高一些;

js和app的交互方案

js和app的交互方案

- 传统方式：url scheme；
- ios: addScriptMessageHandler + Android: webView.addJavascriptInterface

传统方式: 通过scheme url的方式

```
function invoke(action, data, callback) {
  // 拼装 schema 协议,action对应需要实现的方法名
  var schema = "myshema://utils/" + action;

  // 拼接参数 data对应参数
  schema += "?a=a";
  var key;
  for (key in data) {
    if (data.hasOwnProperty(key)) {
      schema += "&" + key + "=" + data[key];
    }
  }

  // 处理 callback
  var callbackName = "";
  if (typeof callback === "string") {
    callbackName = callback;
  } else {
    callbackName = action + Date.now();
    window[callbackName] = callback;
  }
  schema += "&callback=" + callbackName;
  //最终拼接出来应该是zhezhong这种形式 myshema://utils/actioin?a=a&key=value&callback=callbackName
  // 触发
  var iframe = document.createElement("iframe");
  iframe.style.display = "none";
  iframe.src = schema; // 重要! 此处会发送连接,会被webview捕获到
  var body = document.body;
  body.appendChild(iframe);
  setTimeout(function() {
    body.removeChild(iframe);
    iframe = null;
  });
}
```

传统方式: 通过scheme url的方式

缺点:

- 由于请求的url有长度限制，当传递的数据量大时（比如base64字符串），会丢失数据；
- (据说)间隔时间很短的请求可能会只发送一次，因为系统会有拦截请求、解析url等同步操作

新方案:Android端具体实现

```
// 步骤一: 暴露nativeObject对象给webview的js的window对象上
public void updateJsNative() {
    if(Build.VERSION.SDK_INT >= Build.VERSION_CODES.KITKAT&&!AppContext.isRelease()) {
        WebView.setWebContentsDebuggingEnabled(true);
    }
    webView.addJavascriptInterface(new NativeObject4Web(this, webView), "nativeObject");
}


// 步骤二:  定义的api方法必须用JavascriptInterface来装饰, js中才能访问到NativeObject4Web类的方法
@JavascriptInterface
@Override
public void postMessage(String msg) {
    handleMsg(msg);
}


// 步骤三:  安卓执行js回调,使用'loadUrl'方法来执行window对象上的方法
private void responseMsg(String callbackId, ResponseBean responseBean) {
String json = new Gson().toJson(responseBean);
// 安卓 此处应该优化一下, 因为js有传给app端 'callbackName', 此处不应该写死
webView.loadUrl("javascript:window.hxBridge.Core.callbackDispatcher('" + callbackId + "','" + json + "')");
}
```

新方案:IOS端具体实现

IOS端的Webview:

WKWebview: 基于浏览器的webkit内核实现

UIWebview：据称有内存泄漏的问题，比较古老，而且有bug

...

新方案:IOS端具体实现

Instance Method

addScriptMessageHandler:name:

Adds a script message handler.

Declaration

```
- (void)addScriptMessageHandler:(id<WKScriptMessageHandler>
```

Parameters

`scriptMessageHandler`
The message handler to add.

`name`
The name of the message handler.

Discussion

Adding a script message handler with name *name* causes the JavaScript function `window.webkit.messageHandlers.`*name*`.post Message(`*messageBody*`)` to be defined in all frames in all web views that use the user content controller.

Languag
Swift

SDKs
iOS 8.0+
macOS 1

Framew
WebKit

On This
Declarat
Paramet
Discussi

新方案:IOS端具体实现

```objc
#pragma mark - 创建WKUserController
/** 创建初始化WKWebView需要的WKUserController */
-(WKUserContentController *)createWKUserController {
    WKUserContentController *control = [[WKUserContentController alloc] init];
    __weak typeof(self) weakSelf = self;
    [control addScriptMessageHandler:weakSelf name:HXWKWebViewJSBridgeMessageHandlerName];
    return control;
}
```

项目实战中的技术点和问题解决

项目实战中的技术点和问题解决

由于内容较多，查看wiki: http://git.highso.com.cn:81/fe/fe-blog/issues/37 和项目代码进行演示

更优雅的实现方式?

更优雅的实现方式?

由于很多移动端项目在开发过程中都需要hybrid能力，所以抽离成一个npm包:

项目地址: http://git.highso.com.cn:81/fe/fe-common-hybrid
npm仓库地址: http://123.126.133.226:4872/#/detail/fe-common-hybrid

Q&A