通用手机端框架开发文档

一、基本信息：

代码地址：<https://github.com/cn-cerc/summer-swift>

语 言：采用swift语言开发

支持版本：iOS 8.0以上

基本框架：采用苹果最新的WKWebView开发

二、框架介绍：

该框架主要是一个以WKWebView包裹着的浏览器，其中包含了自定义的导航栏、设置页面缩放比例、导航栏返回按钮、推送、支付等原生与js交互的功能。

三、js交互规范：

JS端：

/\*

参数说明

1.type 区分每一个响应事件

2.data js传给手机端的数据

\*/

//JS响应方法列表

//webViewApp swift端的方法名

window.webkit.messageHandlers.webViewApp.postMessage{

type:“”;

data:“”

}

swift端：

方法名：webViewApp

//1. JS调用OC 添加处理脚本

configuretion.userContentController.add(self, name: “webViewApp")

//在代理方法中处理对应事件

func userContentController(\_ userContentController: WKUserContentController, didReceive message: WKScriptMessage) {

//处理返回的数据 ，根据js传过来的type来处理对应的事件

例：

let type = (message.body as! Dictionary<String,String>)["type"]

if type == "类型名" {//

}else if type == "类型名" {//

}

}

//2. native 调用js

/\*

参数说明

1.methodName js方法

2.type string类型

\*/

self.webView.evaluateJavaScript(methodName, completionHandler: { (item:Any?, error:Error?) in

})

四、功能调用规范

1. 修改框架的 MyConfig.swift 文件中的

let URL\_APP\_ROOT = “http://www.baidu.com”//域名地址

2. 替换工程中的欢迎页图片，以欢迎页1，欢迎页2，欢迎页3，欢迎页4 的方式命名

**1)** 微信支付**:(**详细集成文档参考微信开发者平台，此处只描述js与手机端相互处理的过程**)**

JS端：type： WXPay

window.webkit.messageHandlers.webViewApp.postMessage{

type:“WXPay”;

data:“签名后的信息”

}

swift端：

func userContentController(\_ userContentController: WKUserContentController, didReceive message: WKScriptMessage) {

//处理返回的数据 ，根据js传过来的type来处理对应的事件

let type = (message.body as! Dictionary<String,String>)[“type"]

if type == "WXPay" {

let request = PayReq()

request.openID = (message.body as! Dictionary<String,String>)["appid"]

request.nonceStr = (message.body as! Dictionary<String,String>)["nonce\_str"]

request.package = "Sign=WXPay"

request.partnerId = (message.body as! Dictionary<String,String>)["mch\_id"]

request.prepayId = (message.body as! Dictionary<String,String>)["prepay\_id"]

request.timeStamp = UInt32((message.body as! Dictionary<String,String>)["timestamp"]!)!

request.sign = (message.body as! Dictionary<String,String>)["sign"]

WXApi.send(request)

}

}

**2)** 支付宝支付：**(**详细集成文档参考支付宝开发者平台，此处只描述js与手机端相互处理的过程**)**

JS端：

1.详细集成见

https://doc.open.alipay.com/docs/doc.htm?spm=a219a.7629140.0.0.bpU049&treeId=203&articleId=105285&docType=1

swift端：

func webView(\_ webView: WKWebView, didStartProvisionalNavigation navigation: WKNavigation!) {

let orderInfo = AlipaySDK.defaultService().fetchOrderInfo(fromH5PayUrl: webView.url?.absoluteString)

if orderInfo != nil {

AlipaySDK.defaultService().payUrlOrder(orderInfo, fromScheme: "alipay", callback: { (result:[AnyHashable : Any]?) in

if result?["resultCode"] as! String == "9000"{

let urlStr = result?["returnUrl"]

self.loadUrl(urlStr: urlStr as! String)

}else{

if self.webView.canGoBack {

self.webView.goBack()

}

}

})

}

}

步骤一： 实现UIWebViewDelegate协议，拦截H5的URL；

步骤二： 调用SDK提供的“获取订单信息接口(fetchOrderInfoFromH5PayUrl)”对拦截的URL进行处理

特别说明：swift端需要再拦截URL之后重新加载一次才能调起原生支付宝（待解决）

**3)** 极光推送

服务器端：

1.详细集成见极光推送官方文档

链接：<http://docs.jiguang.cn/jpush/server/3rd/java_sdk/>

swift端：

1.在 JPush的管理Portal 上创建应用并上传APNs证书。如果对APNs证书不太了解 请参考： [iOS 证书设置指南](http://docs.jiguang.cn/jpush/client/iOS/ios_cer_guide)

2.创建成功后自动生成 AppKey 用以标识该应用

3.将得到的 appkey 替换框架 MyConfig.swift 文件中的 appkey

**4)** 拨打电话

JS端：type： call

window.webkit.messageHandlers.webViewApp.postMessage{

type:“call”;

data:“电话号码”

}

swift端：

func userContentController(\_ userContentController: WKUserContentController, didReceive message: WKScriptMessage) {

//处理返回的数据 ，根据js传过来的type来处理对应的事件

let type = (message.body as! Dictionary<String,String>)[“type"]

if type == "call" {

let alertController = UIAlertController.init(title: (message.body as! Dictionary<String,String>)["t"], message: nil, preferredStyle: .alert)

let alertAction1 = UIAlertAction.init(title: "取消", style: .cancel) { (action:UIAlertAction) in

}

let alertAction2 = UIAlertAction.init(title: "确定", style: .default) { (action:UIAlertAction) in

let allString = "tel:\(message.body as! Dictionary<String,String>)[“data"]!)"

UIApplication.shared.openURL(NSURL.init(string: allString) as! URL)

}

alertController.addAction(alertAction1)

alertController.addAction(alertAction2)

self.present(alertController, animated: true, completion: nil)

}

}

5)