微信支付 第三篇 微信调用H5页面进行支付

上一篇讲到拿到了 **预支付交易标识 wx251xxxxxxxxxxxxxxxxxxxxxxxxxxxx078700**

第四步，是时候微信内H5调起支付了！

先准备网页端接口请求参数列表

微信文档中已经明确给出了所有参数名和参与签名计算的参数，即

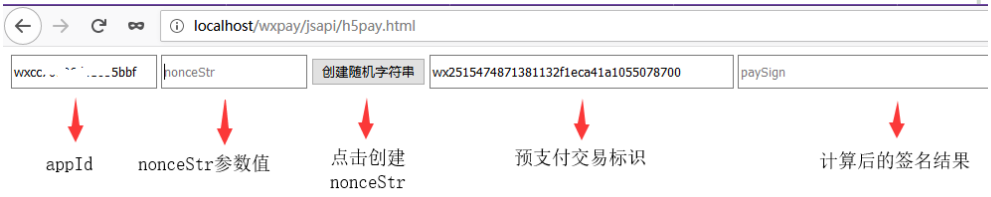
- appId，具有支付权限的与微信支付商户关联绑定的公众号的APPID
- timeStamp，10位当前Unix时间
- nonceStr，<=32 位的随机信息，请求方自行随意生成即可
- package，统一下单接口返回的prepay_id参数值，提交格式为：prepay_id=xxx
- signType，签名方式，**MD5**或**HMAC-SHA256**，默认**MD5**，需要与统一下单的签名类型一致
- paySign，计算后的签名值

计算签名

微信文档中给出参与签名的参数是 **appId、timeStamp、nonceStr、package、signType**，要注意计算签名的方法和《微信支付 第二篇 JSAPI 调用统一下单接口获取预支付交易数据》计算方式是一致的，回顾一下

- 先把参数正序排序
- 参数值为空的去除不参与下面计算
- 把所有符合条件的参数键值对按照URL查询参数格式组成字符串即 **key1=value1&key2=value2...**得到结果字符串 **StringA**
- 把 **StringA** 后面再加上一个参数值对即 **...&key=MCHKEY** 得到结果字符串 **StringB** （MCHKEY为商户密钥）
- 将 **StringB** 按签名方式 signType 进行计算
- 将计算结果转为大写字符串，该值赋值给 paySign

需要注意的是，参数名要严格按照腾讯文档的命名方式，包括大小写，如 **appId** 不能写成 **appid** 或 **appID**。开发时为了方便，H5页面将部分参数设置成了可编辑文本框，先看一下它的样子



本例中，点击创建随机字符串将获得当前时间戳，将其作为 **nonceStr** 和 **timeStamp** 的值，点击创建签名计算出完整结果，**本例中由于 JS 实现 hmacsha256 比较麻烦，就改用了 MD5 方式，根据微信要求统一下单接口中的签名算法也改为了 MD5**，下面列出本例实现代码

```
<!--创建完整签名-->
    <script type="text/javascript">
        function onCreatePaySign() {
            // 从网页的各个文本框获取参数值
            var appid = $("#text-appid").val();
            var nonceStr = $("#text-nonceStr").val();
            var prepayid = $("#text-prepayid").val();
            var timeStamp = nonceStr;
            var signType = "MD5";

            // 拼接字符串
            var sourceStr = "";
            sourceStr += "appId=" + appid;
            sourceStr += "&nonceStr=" + nonceStr;
            sourceStr += "&package=prepay_id=" + prepayid;
            sourceStr += "&signType=" + signType;
            sourceStr += "&timeStamp=" + timeStamp;
            sourceStr += "&key=asxxxxxxxxxxxxxxxN82";

            // MD5方式计算
```

5. #.NET# DataGrid显示大量数据——DataGridView虚模式(2)

```
            var hash = md5(sourceStr).toUpperCase();
            // 在页面中查看一下计算结果
            $("#text-paySign").val(hash);
            // 在控制台查看拼接字符串原文和MD5结果
            console.log("拼接字符串结果: " + sourceStr);
            console.log("MD5结果: " + hash);
        }
    </script>
```

对比查看结果（为了不泄露，appId和mchKey均作了处理，但因为算法固定，因此签名计算结果完全不影响正确性）

真实设备下调用微信支付接口进行实际支付

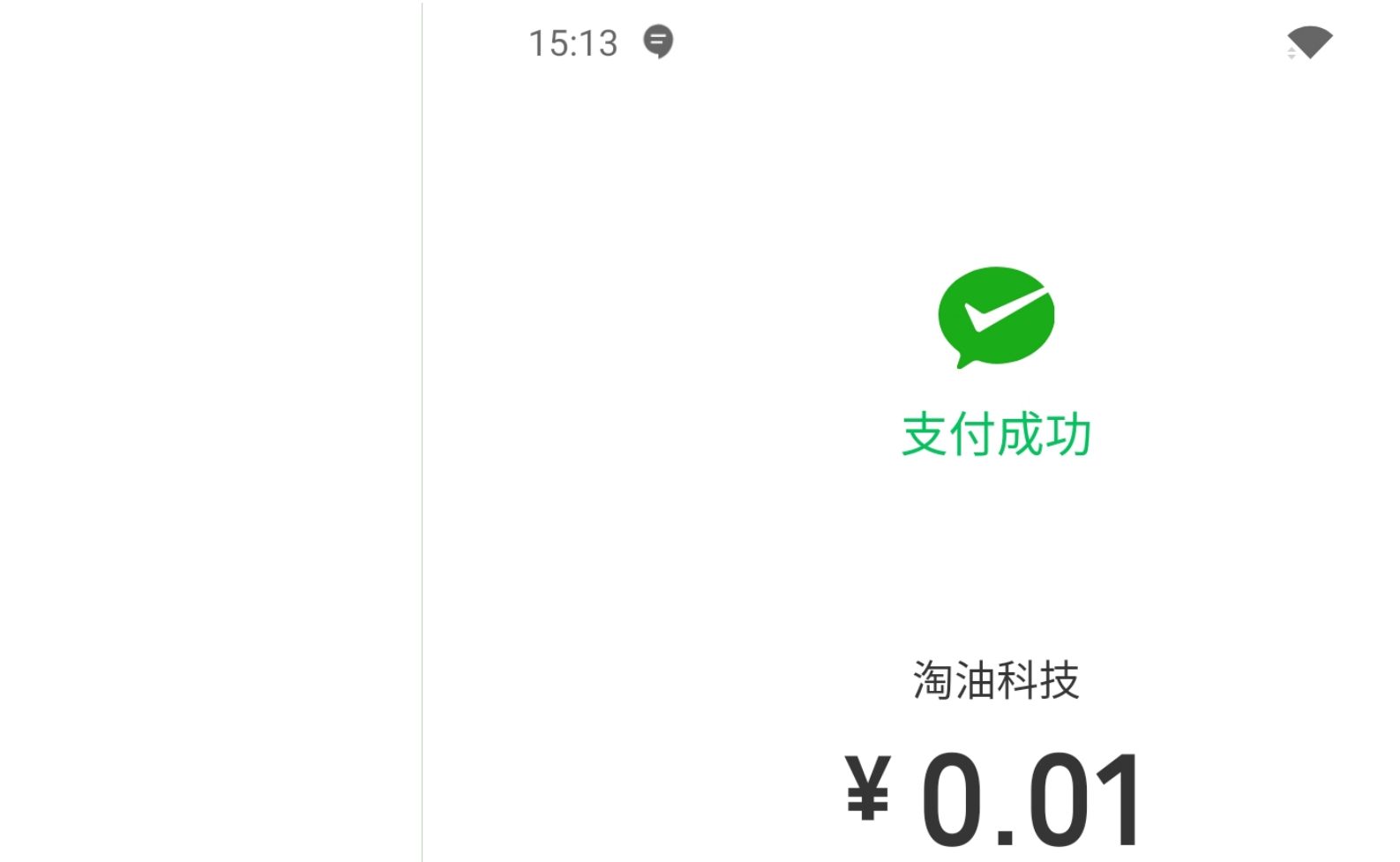
这一步的"坑"是，**必须要在真实设备下进行访问**，也就是手机、平板等运行微信APP的设备，微信开发者工具都不行，会报错无法运行。因此，H5 页面或者说运行微信支付的那部分 JS 一定要在真实设备上才能正确运行。先看一下微信给出的源代码（官方文档里有）

```
function onBridgeReady(){
   WeixinJSBridge.invoke(
      'getBrandWCPayRequest', {
         "appId":"wx2421b1c4370ec43b",     //公众号名称，由商户传入
         "timeStamp":"1395712654",         //时间戳，自1970年以来的秒数
         "nonceStr":"e61463f8efa94090b1f366cccfbbb444", //随机串
         "package":"prepay_id=u802345jgfjsdfgsdg888",
         "signType":"MD5",         //微信签名方式:
         "paySign":"70EA570631E4BB79628FBCA90534C63FF7FADD89" //微信签名
      },
      function(res){
      if(res.err_msg == "get_brand_wcpay_request:ok" ){
      // 使用以上方式判断前端返回,微信团队郑重提示:
            //res.err_msg将在用户支付成功后返回ok，但并不保证它绝对可靠。
      }
   });
}
```

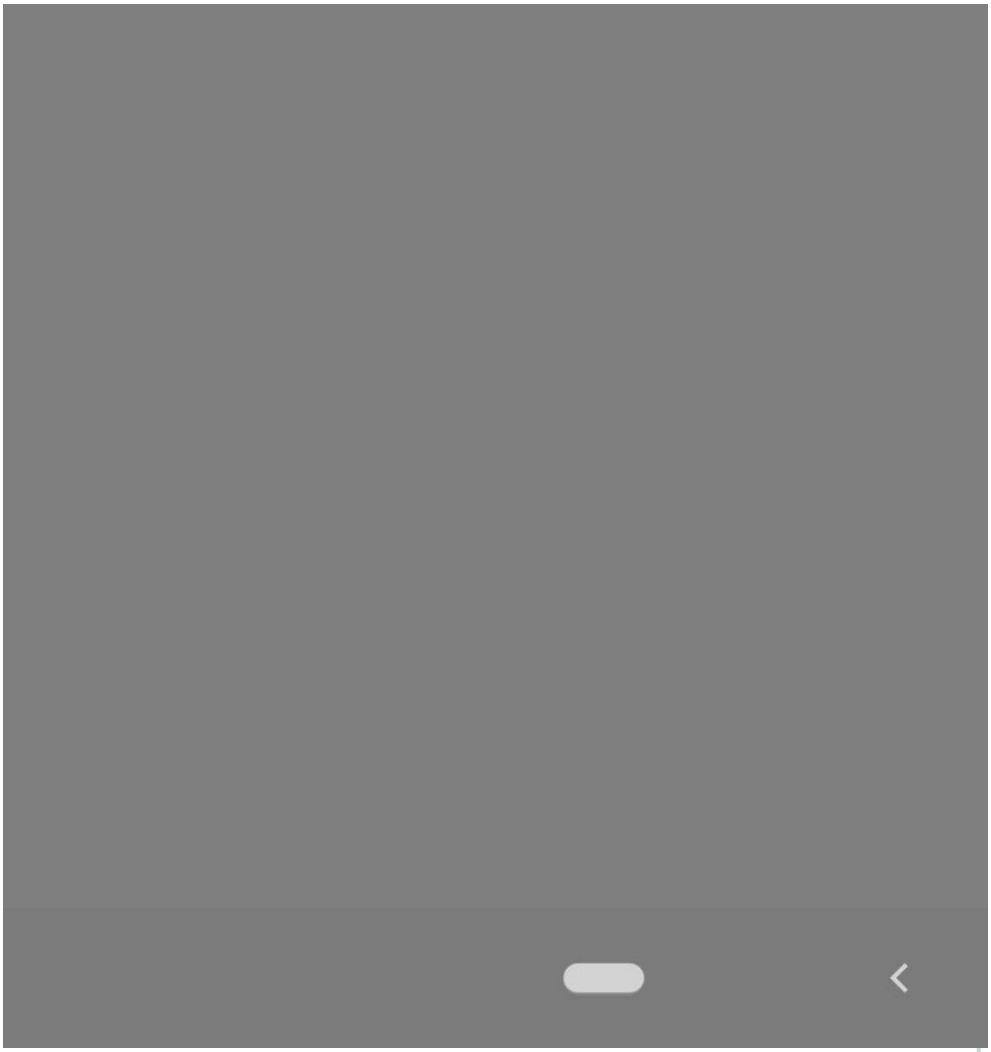
就是这段代码，必须要在真实设备运行，因此本例中采取的方式是将 H5 页面放到服务器上，用开发者工具运行 **获取openid** 和 **获取预支付交易标识** 两个步骤，拿到交易标识后在手机端微信里打开 H5 页面，生成签名并运行支付。根据实际情况，对放到 H5 页面中的 **onBridgeReady** 函数进行了修改，使其从文本框中获取真实数据

```
<!--微信支付调用-->
    <script type="text/javascript">
        function onBridgeReady() {
            WeixinJSBridge.invoke(
                'getBrandWCPayRequest', {
                    "appId": $("#text-appid").val(), //公众号名称，由商户传入
                    "timeStamp": $("#text-nonceStr").val(), //时间戳，自1970年以来的秒数
                    "nonceStr": $("#text-nonceStr").val(), //随机串
                    "package": "prepay_id=" + $("#text-prepayid").val(),
                    "signType": "MD5", //微信签名方式:
                    "paySign": $("#text-paySign").val() //微信签名
                },
                function (res) {
                    console.log("微信支付返回值:");
                    console.log(res);

                    if (res.err_msg == "get_brand_wcpay_request:ok") {
                        // 使用以上方式判断前端返回,微信团队郑重提示:
                        //res.err_msg将在用户支付成功后返回ok，但并不保证它绝对可靠。
                        alert("get_brand_wcpay_request:ok");
                    }

                    alert(res.err_msg);
                    alert(res); // 显示是个 Object
                });
        }
    </script>
```

在手机中，点击页面上的支付按钮，将调起真实的微信支付页面，支付时忘了截图，首先先显示的是信息页和金额，点击支付输入密码/录入指纹，验证成功即支付成功，显示支付成功页面（这个有），并且在用户微信内也会接到来自微信支付的通知消息。本例实现时并没有做H5支付成功后的处理页面，因此在支付成功页面中点击完成，就回到了H5页面中，运行了 **function (res)** 内的代码，并在弹窗中显示出了相关信息


15:13
支付成功
淘油科技
¥0.01


完成
15:13
微信支付 JSAPI 方式支付示例
bf
1561533142
创建
203B50CCC6D72F75FE0677D63
创建签名
支付
get_brand_wcpay_request:ok
确定



总结

至此，完成了微信支付 JSAPI 支付主要流程。罗列一下关键点

- 服务器环境使用了自有真实网站 wxpay.txxxt.com，完整路径 **http://wxpay.txxxt.com/jsapi/**，这也是在商户管理后台的 **支付授权目录** 中配置的路径，即发起微信支付调用的页面是来自该目录
- 所有页面和接口都放在了该目录下，**getCode.php** + **codeCallback.php** 配合获取 openid，**unifiedorder.php?openid=xxxx** 负责根据openid获取预支付交易标识，拿到交易标识后 **h5pay.html** 负责调起支付
- 前两项可以在web开发者工具内运行，选择公众号网页调试即可，调起支付必须在真实设备运行

不重要的其实最重要

Cinlap Soft
关注 - 18
粉丝 - 10
+加关注

« 上一篇: 微信支付 第二篇 JSAPI 调用统一下单接口获取预支付交易数据

posted @ 2019-06-26 18:25 Cinlap Soft 阅读(99) 评论(0) 编辑 收藏

相关博文:
· web页面调用支付宝支付
· 微信支付-公众号支付H5调用支付详解
· 微信支付开发h5调用
· 微信公众号支付（三）：页面调用微信支付JS并完成支付
· 【原创分享·支付宝支付】HBuilder打包APP调用支付宝客户端支付