

# [接口测试用例-编写指导手册]

使用 Excel 完成 Http 协议的单或多接口自动化操作

[宝付网络(上海)科技有限公司]  
[上海市 浦东新区 居里路 99 号]  
[17097159420]  
[Early@baofoo.com]  
2016 年 9 月 21 日

# 目 录

引： .....	3
使用指南： .....	3
第一章：基础入门了解篇.....	4
1.接口测试环境介绍.....	4
2.接口测试集介绍.....	4
3.接口测试用例介绍.....	5
第二章：接口测试用例编写---基本编写.....	6
1.请求参数下层模板的讲解： .....	6
2.响应参数的讲解： .....	7
第三章：接口测试用例编写---用例编写进阶.....	8
☆☆☆☆☆进阶的写法一： .....	8
9 月 21 更新新增： .....	10
☆☆☆☆☆进阶的写法二： .....	10
新增字段.....	10

# 引：

## 使用指南：

- 1.下载：Update.exe 文件到任意文件夹。
- 2.双击 Update.exe 文件，更新最新的程序（当前目录下下载出：Api\_Auto.exe 文件）
- 3.（首次使用）先双击执行 Api\_Auto.exe （需要先登录，木有账号，找吴宇） ，将会创建对应的 Interface 文件夹。
- 4.Interface 文件夹下 TestCase 文件夹为测试用例存储位置
- 5.Interface 文件夹下 Baofoo\_Rsa 文件夹为外部其他文件存储位置，例如：公私钥加密文件、用例所需的大数据文件。
- 6.log 会输出在 Interface 文件夹中，提供分析问题。

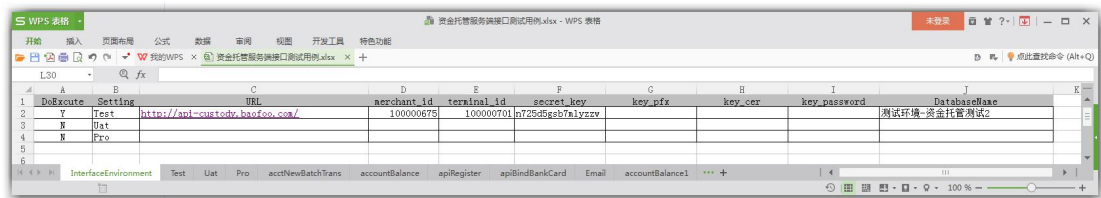
有关关键字，可以访问作者 blog.



<http://early.farbox.com/>

# 第一章：基础入门了解篇

## 1.接口测试环境介绍



表名: **InterfaceEnvironment**, 这个表可以设置接口执行的环境, 当前文档只展示了测试环境的接口。

列名: **DoExcute**: 指定是否执行当前环境, Y: 执行, N 不执行。

列名: **Setring**: 指定了当前环境的测试集的存储位置。

列名: **URL**: 存放的是当前环境的域名。

列名: **merchant\_id**: 存放当前商户的商户号。

列名: **terminal\_id**: 存放当前商户的终端号。

列名: **secret\_key**: 存放当前商户的 MD5 秘钥。

列名: **key\_pfx**: 存放当前商户的公钥文件名称。

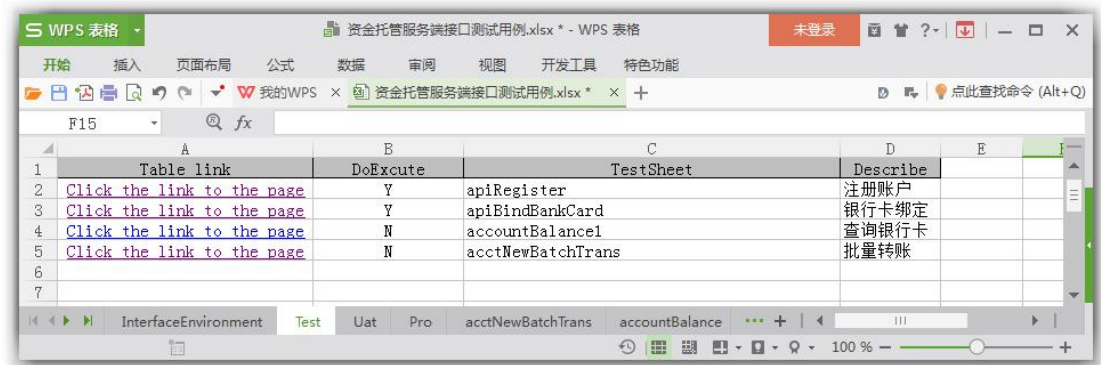
列名: **key\_cer**: 存放当前商户的私钥文件名称。

列名: **key\_password**: 存放当前商户的私钥文件密码。

列名: **DatabaseName**: 目前无用, 后续扩展数据库使用。

以上就接口测试环境的介绍, 基本的参数, 找开发要来填充即可。

## 2.接口测试集介绍



表名: **Test**: 测试环境的测试集, 对应之前测试环境的 **Setting** 指定的测试集 (准生产环境: Uat, 生产环境: Pro)

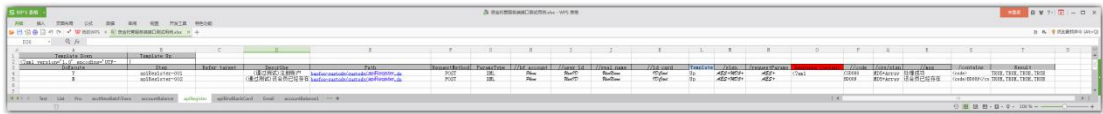
列名: **Table link**: 超链接, 用于接口测试用例过多时快速跳转至对应测试用例。

列名: **DoExcute**: 与测试环境的 **DoExcute** 相同, 表示当前测试用例是否执行。

列名: **TestSheet**: 接口测试用例的存放表, 内容与表名相同, 表示用例在对应表单。 (注: 一定需要是一致的。)

列名: **Describe**: 描述。描述当前接口的作用, 查看日志是可用于参考。

3.接口测试用例介绍



先来个大致介绍：  
表名：**apiRegister**：这是一个注册接口。  
列名：**Template\_Down**：下层接口模板。  
列名：**Template\_Up**：上层接口模板。  
先介绍下这两个接口模板吧。  
由于公司的接口参数基本都是：一层键值对中包含一个 XML 格式的参数，但是由于 XML 格式不支持有 2 个头文件，导致程序解析时无法解析，所以就将参数分为 2 层。（注：Json 格式可以包含，但资金托管暂不支持 json）

情况就是这样，下面可以看一下模板信息：

Template_Down	Template_Up
<pre>&lt;?xml version='1.0' encoding='UTF-8'?&gt; &lt;custody_req&gt;   &lt;bf_account&gt;15000918908&lt;/bf_account&gt;   &lt;user_id&gt;20140375&lt;/user_id&gt;   &lt;real_name&gt;李清照&lt;/real_name&gt;   &lt;id_card&gt;32030019800918181X&lt;/id_card&gt;   &lt;account_type&gt;1&lt;/account_type&gt; &lt;/custody_req&gt;</pre>	

详细情况，可参看【宝付资金托管商户接口文档(房天下定制版).doc】里面的 apiRegister 接口。

- 继续...
- 列名：**DoExcute**: 表示是否执行，Y：执行，N 不执行。
  - 列名：**Step**: 步骤名称，与测试用例一样，必须唯一。
  - 列名：**Refer target**: 目标测试步骤，可不填，进阶多接口测试用例编写可讲。
  - 列名：**Describe**: 描述，描述当前测试步骤的作用。
  - 列名：**Path**: 路径，存放 Http 接口域名以后的路径，用于请求。
  - 列名：**RequestMethod**: 接口请求方式：POST，目前只支持 POST，后续可扩展。
  - 列名：**ParamsType**: 下层参数模板请求时的类型，支持 Json、Xml
  - 列名：.... 下层参数的参数操作
  - 列名：**Template**: 切换模板，从下层模板的切换至上层模板。
  - 列名：.... 上层模板的参数操作
  - 列名：**Response\_Content**: 不用填写值，系统会写入响应报文并切换至响应报文
  - 列名：.... 响应报文的断言操作
  - 列名：**Result**: 最终断言，判断当前测试用例是否执行成功。

接口测试用例介绍结束，现在进入学习编写接口测试用例吧。

第二章：接口测试用例编写---基本编写

```
{
  "custody_req": {
    "A": [
      "a",
      "a",
      "a",
      "a"
    ],
    "test": "123",
    "num": [
      {
        "A": [
          "a",
          "a",
          "a",
          "a"
        ],
        "B": [
          "b",
          "b1",
          "b",
          "b",
          "b",
          "b"
        ]
      }
    ],
    "B": "b"
  },
  {
    "C": "c",
    "D": "d"
  },
  {
    "E": "e",
    "F": "f"
  }
],
"tester": {
  "TEST": "123",
  "t0": {
    "TEST": "123"
  }
}
}
```

←Json格式的参数字模板

↓Xml格式的参数字模板

```
<?xml version="1.0" encoding="UTF-8" ?>
<custody_req>
  <A>a</A>
  <A>a</A>
  <A>a</A>
  <A>a</A>
  <num>
    <A>a</A>
    <A>a</A>
    <A>a</A>
    <A>a</A>
    <A>
      <B>b</B>
      <B>b1</B>
      <B>b</B>
      <B>b</B>
      <B>b</B>
      <B>b</B>
    </A>
    <B>b</B>
  </num>
  <C>c</C>
  <D>d</D>
</num>
  <test>123</test>
  <tester>
    <TEST>123</TEST>
    <t0>
      <TEST>123</TEST>
    </t0>
  </tester>
  <num>
    <E>e</E>
    <F>f</F>
  </num>
</custody_req>
```

讲解的问题是：如何寻址？

模板参数寻址要点：  
/ : 表示绝对路径  
// : 表示相对路径  
\*[] : 此为数值存放时使用，例如\*[1]

注：1. 所有下标数值都是从0开始数  
2. Xml与Json的寻址方式已统一，便于编写寻址路径

下面是栗子：别客气，多唠唠。

1. 编写寻址值为：b1 的绝对路径  
路径：/custody\_req/num/\*[0]/A/\*[5]/B/\*[1]  
结果：b1

2. 编写寻址值为：b1 的相对路径  
1) 路径：//custody\_req/num//B/\*[1]  
2) 路径：//num/\*[0]/A/\*[5]/B/\*[1]  
1) 结果：b1  
2) 结果：b1

3. 编写寻址值为：c 的相对路径  
1) 路径：//custody\_req/C  
2) 路径：//custody\_req//C  
3) 路径：//C  
1) 结果：c  
2) 结果：c  
3) 结果：c

4. 编写寻址值为：123 的相对路径  
1) 路径：//TEST  
2) 路径：//t0/TEST  
1) 结果：123, 123  
2) 结果：123

根据以上的路径寻址，相信大家都知道如何进行寻址，也知道了 Excel 的列名带有//或/就知道是做什么用的了吧？不知道？好吧，是用于寻址。

1.请求参数下层模板的讲解：

RequestMethod	ParamsType	//bf_account	//user_id	//real_name	//id_card	Template	/sign	/requestParams	Response Content
POST	XML	13754854569	123456	流氓兔	430578*****8545	0p	AES+MD5+	AES+	<?xml
POST	XML	13754854569	123456	流氓兔	430578*****8545	0p	AES+MD5+	AES+	

上图，被选中的单元格中的值为要替换的值。例如：

列名：//bf\_account ：将下层模板中指定的值替换成：13754854569

列名：//user\_id ：将下层模板中指定的值替换成：123456



列名: **//real\_name** : 将下层模板中指定的值替换成: 流氓兔

列名: **//id\_card** : 将下层模板中指定的值替换成: 430578\*\*\*\*\*8545

一、请求参数上层模板的讲解:

列名: **Template** 之后就是切换成上层模板了。



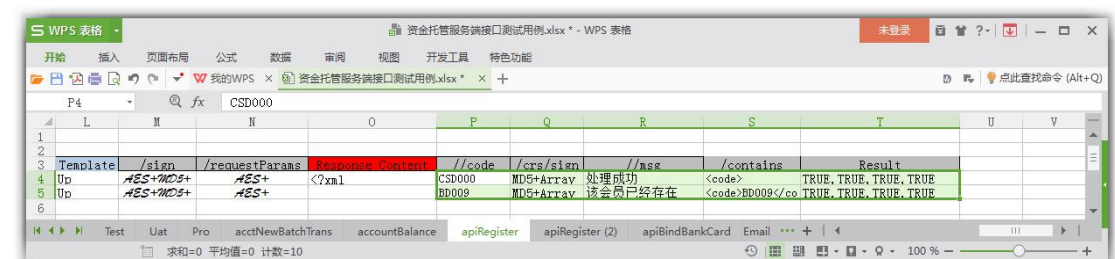
列名: **/sign** : 上层模板对下层模板的操作, 如 AES+MD5+

列名: **/requestParams** : 上层模板对下层模板的操作, 如 AES+  
其实将下层模板参数加密后, 再替换给指定的值。

详细加密关键字, 可以了解作者 blog:[接口测试用例编写指南](#)

## 2. 响应参数的讲解:

列名: **Response\_Content** 之后就是切换成响应报文模板了。



列名: **//code**: 断言报文中 code 值为 “CSD000”。

列名: **/crs/sign**: 断言报文中 sign 值为 MD5+Array 方式加密后的内容。

列名: **//msg**: 断言报文中 msg 值为 “处理成功”。

列名: **/contains**: 断言报文中包含值为 “<msg>处理成功</msg>”

以上响应报文的断言, 正确返回: TURE, 错误返回: FALSE

列名: **Result**: 最终断言! 校验以上报文的实际执行结果与预期结果是否一致, 一致则测试通过, 否则测试失败。

注: 请求报文、响应报文的单元格数量并不限制, 可以根据测试用例的想法, 字段的长短以及不填的效果来组装测试用例。

以上便是接口测试用例的编写入门! 你会了吗? 会了就赶紧编写一个吧, 毕竟书读百遍, 不如手写一篇!

### 第三章：接口测试用例编写---用例编写进阶

☆☆☆☆☆进阶的写法一：

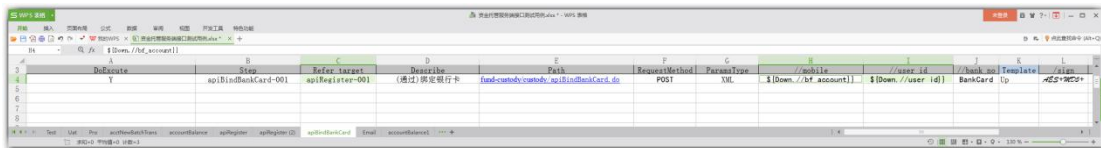


RequestMethod	ParamsType	//bf_account	//user_id	//real_name	//id_card	Template	/sign	/requestParams	ResponseContent
POST	XML	Phone	NewID	NewName	IDCard	Up	AES+MD5+	AES+	<?xml

- 列名：**//bf\_account**：将下层模板中指定的值替换成：Phone 关键字随机生成的手机号码。
- 列名：**//user\_id**：将下层模板中指定的值替换成：UserID 关键字随机生成的 user\_id。
- 列名：**//real\_name**：将下层模板中指定的值替换成：UserName 关键字随机生成的用户姓名。
- 列名：**//id\_card**：将下层模板中指定的值替换成：IDCard 关键字随机生成的身份证号码。

以上为随机生成对应的字段快速执行测试，但是有些测试场景并不适合，例如，单一注册，此方式即可满足，但是再加上绑卡操作。那么就需要获取以上的字段，例如：**user\_id**、**bf\_account**，但上面是随机生成的字段，在未生成前，程序都不知道是什么，怎么办呢？

看下面，参数填充的进阶写法：↓



Step	Refer target	Describe	Path	RequestMethod	ParamsType	/mobile	//user_id	//bank_no	Template	/sign
1	apiBindBankCard-001	apiRegister-001	(通过)绑定银行卡	POST	XML	\${Down.//bf_account}	\${Down.//user_id}	BankCard	Up	AES+MD5+

- 列名：**Refer target**：指定参数获取的目标，填写需要获取接口的唯一值：Step
- 列名：**//user\_id**：此处填写\${Down.//user\_id}
- 列名：**//real\_name**：此处填写\${Down.//user\_id}
- 列名：**//id\_card**：将下层模板中指定的值替换成：BankCard 关键字随机生成的身份证号码。

要点：

0.在 **Refer target** 指定上一接口步骤。

获取值的方式为：

- 1.使用 **\${}** 将参数包起来。



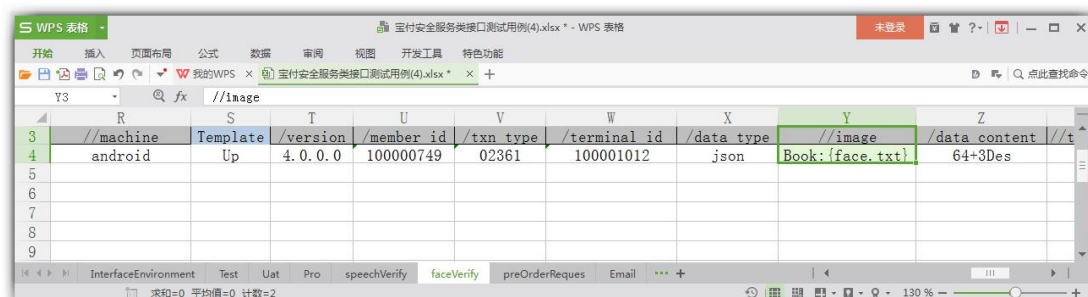
- 2.内部的值可以获取指定模板的值：Up、Down、Response
- 3.使用.将分隔模板和参数路径
- 4.指定对应值的路径：//user\_id
- 5.可以只写路径,不写模板，但不建议这样做。

下面可以看下进阶用例的执行日志：

```
08-31 17:27:17 [3437] [com.Api.Utils.ExcelUtils] [151] - 开始执行测试用例
08-31 17:27:19 [5505] [com.Api.Utils.ExcelUtils] [295] - 执行步骤：apiRegister-001
08-31 17:27:19 [5505] [com.Api.Utils.ExcelUtils] [296] - 执行步骤的描述：(通过测试)注册账户
08-31 17:27:19 [5515] [com.Api.Utils.ExcelUtils] [189] - 替换参数：key=//bf_account; value=18787773164
08-31 17:27:19 [5522] [com.Api.Utils.ExcelUtils] [189] - 替换参数：key=//user_id; value=6637640
08-31 17:27:19 [5523] [com.Api.Utils.ExcelUtils] [189] - 替换参数：key=//real_name; value=师伯融
08-31 17:27:19 [5528] [com.Api.Utils.ExcelUtils] [189] - 替换参数：key=//id_card; value=620102196009118867
08-31 17:27:19 [5955] [com.Api.Utils.ExcelUtils] [189] - 替换参数：key=//sign; value=5eb03246324f3211e5e170fd0a3b2
08-31 17:27:19 [5957] [com.Api.Utils.ExcelUtils] [189] - 替换参数：key=//requestParams; value=412284fac44951266706558412a5511c689c5538754005b43c3dab06f05bb0cf7c
08-31 17:27:19 [5957] [com.Api.Utils.ExcelUtils] [227] - 输出请求参数：sign=5eb03246324f3211e5e170fd0a3b2&requestParams=412284fac44951266706558412a5511c689c
08-31 17:27:19 [5959] [com.Api.Utils.ExcelUtils] [228] - 请求地址：http://api-custody.baofoo.com/api-custody/custody/apiRegister.do
08-31 17:27:20 [7103] [com.baofoo.util.HttpRequestUtil] [150] - 状态码：200
08-31 17:27:20 [7104] [com.Api.Utils.ExcelUtils] [237] - 输出响应参数：<?xml version="1.0" encoding="UTF-8" ?><rs><code>CSD000</code><msg>处理成功</msg><sign>5
08-31 17:27:20 [7107] [com.Api.Utils.ExcelUtils] [461] - 获取实际值：CSD000
08-31 17:27:20 [7107] [com.Api.Utils.ExcelUtils] [462] - 获取预期值：CSD000
08-31 17:27:20 [7107] [com.Api.Utils.ExcelUtils] [201] - 获取执行结果：TRUE,
08-31 17:27:20 [7107] [com.Api.Utils.ExcelUtils] [482] - 匹配参数，加密参数，动态匹配：MD5+Array
08-31 17:27:20 [7109] [com.Api.Config.ResponseMethod] [93] - 获取实际值：e89e3be8e509b8a220c1f3ca3490b089
08-31 17:27:20 [7109] [com.Api.Config.ResponseMethod] [94] - 获取预期值：e89e3be8e509b8a220c1f3ca3490b089
08-31 17:27:20 [7109] [com.Api.Utils.ExcelUtils] [201] - 获取执行结果：TRUE, TRUE,
08-31 17:27:20 [7110] [com.Api.Utils.ExcelUtils] [461] - 获取实际值：处理成功
08-31 17:27:20 [7111] [com.Api.Utils.ExcelUtils] [462] - 获取预期值：处理成功
08-31 17:27:20 [7111] [com.Api.Utils.ExcelUtils] [201] - 获取执行结果：TRUE, TRUE, TRUE,
08-31 17:27:20 [7111] [com.Api.Utils.ExcelUtils] [482] - 匹配参数，加密参数，动态匹配：/Contains
08-31 17:27:20 [7112] [com.Api.Utils.ExcelUtils] [201] - 获取执行结果：TRUE, TRUE, TRUE, TRUE,
08-31 17:27:20 [7112] [com.Api.Utils.ExcelUtils] [314] - 执行步骤：apiRegister-001--->执行结果值：TRUE, TRUE, TRUE, TRUE
08-31 17:27:20 [7112] [com.Api.Utils.ExcelUtils] [315] - 执行步骤：apiRegister-001--->预期结果值：TRUE, TRUE, TRUE, TRUE
08-31 17:27:20 [7112] [com.Api.Utils.ExcelUtils] [320] - 最终执行结果：通过
08-31 17:27:27 [14203] [com.Api.Utils.ExcelUtils] [151] - 开始执行测试用例
08-31 17:27:31 [17448] [com.Api.Utils.ExcelUtils] [295] - 执行步骤：apiBindBankCard-001
08-31 17:27:31 [17448] [com.Api.Utils.ExcelUtils] [296] - 执行步骤的描述：(通过)绑定银行卡
08-31 17:27:31 [17448] [com.Api.notes.FindAndSaveApiValue] [47] - 获取上一接口执行结果：==>【Pass】
08-31 17:27:31 [17449] [com.Api.notes.FindAndSaveApiValue] [179] - 获取上一接口参数信息：<?xml version="1.0" encoding="UTF-8"?>
<custody_req><bf_account>18787773164</bf_account><user_id>6637640</user_id><real_name>师伯融</real_name><id_card>620102196009118867</id_card><account_type>1</acc
08-31 17:27:31 [17451] [com.Api.Utils.ExcelUtils] [189] - 替换参数：key=//mobile; value=18787773164
08-31 17:27:31 [17452] [com.Api.notes.FindAndSaveApiValue] [47] - 获取上一接口执行结果：==>【Pass】
08-31 17:27:31 [17452] [com.Api.notes.FindAndSaveApiValue] [179] - 获取上一接口参数信息：<?xml version="1.0" encoding="UTF-8"?>
<custody_req><bf_account>18787773164</bf_account><user_id>6637640</user_id><real_name>师伯融</real_name><id_card>620102196009118867</id_card><account_type>1</acc
08-31 17:27:31 [17454] [com.Api.Utils.ExcelUtils] [189] - 替换参数：key=//user_id; value=6637640
08-31 17:27:31 [17458] [com.Api.Utils.ExcelUtils] [189] - 替换参数：key=//bank_no; value=5264108397821514
08-31 17:27:31 [17461] [com.Api.Utils.ExcelUtils] [189] - 替换参数：key=//sign; value=b6e929c368f9ee38a27eec32a7fb2088
08-31 17:27:31 [17462] [com.Api.Utils.ExcelUtils] [189] - 替换参数：key=//requestParams; value=412284fac44951266706558412a5511c689c5538754005b43c3dab06f05bb0cf7c
08-31 17:27:31 [17462] [com.Api.Utils.ExcelUtils] [227] - 输出请求参数：sign=b6e929c368f9ee38a27eec32a7fb2088&requestParams=412284fac44951266706558412a5511c689c
08-31 17:27:31 [17462] [com.Api.Utils.ExcelUtils] [228] - 请求地址：http://api-custody.baofoo.com/fund-custody/custody/apiBindBankCard.do
08-31 17:27:31 [18010] [com.baofoo.util.HttpRequestUtil] [150] - 状态码：200
08-31 17:27:31 [18010] [com.Api.Utils.ExcelUtils] [237] - 输出响应参数：<?xml version="1.0" encoding="UTF-8" ?><rs><code>CSD000</code><msg>处理成功</msg><card_
08-31 17:27:31 [18011] [com.Api.Utils.ExcelUtils] [461] - 获取实际值：CSD000
08-31 17:27:31 [18012] [com.Api.Utils.ExcelUtils] [462] - 获取预期值：CSD000
08-31 17:27:31 [18012] [com.Api.Utils.ExcelUtils] [201] - 获取执行结果：TRUE,
08-31 17:27:31 [18012] [com.Api.Utils.ExcelUtils] [482] - 匹配参数，加密参数，动态匹配：MD5+Appoint
08-31 17:27:31 [18012] [com.Api.Config.ResponseMethod] [111] - 获取实际值：e89e3be8e509b8a220c1f3ca3490b089
08-31 17:27:31 [18012] [com.Api.Config.ResponseMethod] [112] - 获取预期值：e89e3be8e509b8a220c1f3ca3490b089
08-31 17:27:31 [18013] [com.Api.Utils.ExcelUtils] [201] - 获取执行结果：TRUE, TRUE,
08-31 17:27:31 [18013] [com.Api.Utils.ExcelUtils] [461] - 获取实际值：处理成功
08-31 17:27:31 [18013] [com.Api.Utils.ExcelUtils] [462] - 获取预期值：处理成功
08-31 17:27:31 [18013] [com.Api.Utils.ExcelUtils] [201] - 获取执行结果：TRUE, TRUE, TRUE,
08-31 17:27:31 [18013] [com.Api.Utils.ExcelUtils] [482] - 匹配参数，加密参数，动态匹配：/Contains
08-31 17:27:31 [18013] [com.Api.Utils.ExcelUtils] [201] - 获取执行结果：TRUE, TRUE, TRUE, TRUE,
08-31 17:27:31 [18014] [com.Api.Utils.ExcelUtils] [314] - 执行步骤：apiBindBankCard-001--->执行结果值：TRUE, TRUE, TRUE, TRUE
08-31 17:27:31 [18014] [com.Api.Utils.ExcelUtils] [315] - 执行步骤：apiBindBankCard-001--->预期结果值：TRUE, TRUE, TRUE, TRUE
08-31 17:27:31 [18014] [com.Api.Utils.ExcelUtils] [320] - 最终执行结果：通过
08-31 17:27:31 [18168] [com.Api.Email.SendEmail] [57] - Report Link：暂时还没有web端
08-31 17:27:32 [18681] [com.Api.Email.SendEmail] [260] - mail send success!
```

# 9月21更新新增:

## ☆☆☆☆☆进阶的写法二:



	R	S	T	U	V	W	X	Y	Z
3	//machine	Template	/version	/member id	/txn type	/terminal id	/data type	//image	/data content
4	android	Up	4.0.0.0	100000749	02361	100001012	json	Book:{face.txt}	64+3Des
5									
6									
7									
8									
9									

列名: `//image` : 此处填写 `Book:{face.txt}`

## 使用场景:

Excel 单元格存储情况:

单元格内容(文本)的长度 32,767 个字符。单元格中只能显示 1,024 个字符;而编辑栏中可以显示全部 32,767 个字符。

这种情况下,如果有大文本数据需要请求,无法存放在单元格内,例如人脸(字符串长度: 50 万)、声纹(字符串长度: 3.6 万)等接口,需要大量数据,这种情况,采用外部文件进行存储。

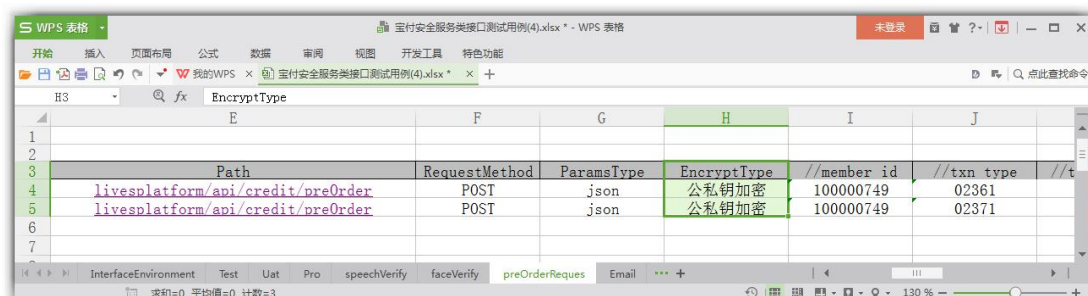
## 要点:

- 1.使用 `Book: {}` 表示为外部文件,将文件名称包起来。
- 2.内部的值为 文件名称(带后缀名),例如:`Book: {face.txt}`
- 3.此文件存储在当前项目 `Interface\Baofoo_Rsa` 路径下

## 特别声明:

由于有大数据的情况存在,所以某一参数长度大于 5000 的 value,将不做存储。多接口的情况,无法获取到其他接口的大数据。

## 新增字段



	E	F	G	H	I	J
3	Path	RequestMethod	ParamsType	EncryptType	//member id	//txn type
4	<code>livesplatform/api/credit/preOrder</code>	POST	json	公私钥加密	100000749	02361
5	<code>livesplatform/api/credit/preOrder</code>	POST	json	公私钥加密	100000749	02371
6						
7						

用例新增字段: `EncryptType` : 加密方式

因加密方式过多,所以特意增加,要求使用公私钥加密时,需要填充:“公私钥加密”

可见上图。