 【**保密**】

**CPIC B2B 业务服务总线平台**



客户端调用示例说明

|  |  |  |  |
| --- | --- | --- | --- |
| 文档状态 | [√] 初稿[ ] 评审通过[ ] 修改[ ] 发布[ ] 作废 | 项目名称 | CPIC B2B业务服务总线平台 |
| 文档标识 | CPICPRJ-DE-003 |
| 当前版本 |  |
| 作 者 |  |
| 部门**/**厂商 |  |
| 完成日期 |  |

CPIC软件工程文档

【内部资料 不得外泄】

**文档版本历史表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 作者 | 操作 | 日期 | 说明 |
| V0.1 | 杜威 | 创建文档 | 2013-12-05 | 初稿 |
|  |  |  |  |  |
|  |  |  |  |  |

**文档审核记录表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 审核人 | 审核人签字 | 审核日期 | 说明 |
|  |  |  |  |  |

**内容范围**

本文档是关于第三方系统调用中国太平洋保险（集团）股份有限公司业务服务总线平台示例说明书。

**适用的对象**

本文档仅适用于中国太平洋保险（集团）股份有限公司B2B业务服务总线项目组以及需要跟CPIC实现系统对接的第三方合作伙伴阅读。

# 编写目的

为第三方系统调用中国太平洋保险（集团）股份有限公司业务服务总线平台提供开发参考，不作为必须遵循的标准，第三方系统可以有自己的实现方式。

# 测试环境信息

开发环境接口地址：[*https://jttpitxtest.cpic.com.cn/jttpitx/itxsvc/param*](https://jttpitxtest.cpic.com.cn/jttpitx/itxsvc/param)

集成环境接口地址：[*https://jttpitxsit.cpic.com.cn/jttpitx/itxsvc/param*](https://jttpitxsit.cpic.com.cn/jttpitx/itxsvc/param)

生产环境接口地址：<https://jttp.cpic.com.cn:8443/jttpitx/itxsvc/param>(生产环境因有白名单，需合作伙伴提前提供出口IP给接口对接人)

# 示例引用到的Jar包

本文档中的实例都是基于Java平台开发的示例，示例将引用到的Jar包如下：



# 调用示例

## 示例代码

**import** java.util.ArrayList;

**import** java.util.List;

**import** org.apache.http.HttpEntity;

**import** org.apache.http.HttpResponse;

**import** org.apache.http.NameValuePair;

**import** org.apache.http.client.HttpClient;

**import** org.apache.http.client.entity.UrlEncodedFormEntity;

**import** org.apache.http.client.methods.HttpPost;

**import** org.apache.http.impl.client.DefaultHttpClient;

**import** org.apache.http.message.BasicNameValuePair;

**import** org.apache.http.params.HttpConnectionParams;

**import** org.apache.http.util.EntityUtils;

**public** **class** HttpsRequestDemo {

 // 太保的测试环境地址

 **private** String targetURL = "https://jttpitxtest.cpic.com.cn/jttpitx/itxsvc/param";

 // 版本标识，固定值 3

请填写太保提供的商户代码，向太保方索取。XML中的partnerCode也于此一致，注意大写。

 **private** String messageRouter = "3";

 // 合作伙伴编码，由太保指定，每个商户不一样

 **private** String partnerCode = "QOROS";

 // 业务协议，固定值 CPIC\_ECOM

 **private** String documentProtocol = "CPIC\_ECOM";

 /\*\*

 \* **@param** args

 \*/

 **public** **static** **void** main(String[] args) {

 HttpsRequestDemo requestDemo = **new** HttpsRequestDemo();

 // xml请求报文，详细的报文结构请考各交易的详细报文示例

 String requestMsg = "";

 **try** {

 String result = requestDemo.sentHttpPostRequest(requestMsg);

 System.*out*.println(result);

 } **catch** (Exception e) {

 e.printStackTrace();

 }

 }

 **private** String sentHttpPostRequest(String requestMsg) **throws** Exception {

 HttpClient httpclient = **new** DefaultHttpClient();

 // 设置超时时间

 **int** timeout = 60000;

 HttpConnectionParams.*setSoTimeout*(httpclient.getParams(), timeout);

 // 注意：必须以post方式发送请求

 HttpPost httppost = **new** HttpPost(targetURL);

 // 设置请求参数

 List<NameValuePair> params = **new** ArrayList<NameValuePair>();

 // 版本标识

 params.add(**new** BasicNameValuePair("messageRouter", messageRouter));

 // 业务伙伴代码

 params.add(**new** BasicNameValuePair("tradingPartner", partnerCode));

 // 业务协议

 params.add(**new** BasicNameValuePair("documentProtocol", documentProtocol));

 // xml请求报文

 params.add(**new** BasicNameValuePair("requestMessage", requestMsg));

 // 注意：编码必须是UTF-8

 HttpEntity request = **new** UrlEncodedFormEntity(params, "UTF-8");

 httppost.setEntity(request);

 // 返回内容为xml，请使用xml解析工具对返回内容进行解析

 HttpResponse httpResponse = httpclient.execute(httppost);

 HttpEntity entity = httpResponse.getEntity();

 String result = **null**;

 **if** (entity != **null**) {

 result = EntityUtils.*toString*(entity);

 }

 **return** result;

 }

}

## 4.2样例报文

<request>

 <head>

 <!-- 商户代码，由太保开发人员提供 -->

 <partnerCode>SZTN</partnerCode>

 <!-- 接口代码，由太保开发人员提供 -->

 <transactionCode>108001</transactionCode>

 <!-- 流水号 -->

 <messageId>2016120800000052</messageId>

 <!-- 有效时间，当天日期，格式严格参照 -->

 <transactionEffectiveDate>2016-12-08 14:04:22

 </transactionEffectiveDate>

 <!-- 商户代码，同partnerCode -->

 <user>SZTN</user>

 <!-- 商户密码，由太保开发人员提供 -->

 <password>+kp0LXaWRRWq2bsb/EJR22O2aqjghfCrGe/OaJJheaJttbKcDxlXiDYfpcxLK1fKm75rGAjtNsTFuxt/v/F+Vw==

 </password>

 </head>

 <body>

 <!-- 双方约定的报文格式 -->

 </body>

</request>

## 4.3常见错误

### 4.3.1 dom4j解析报文失败

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<Values version=*"2.0"*>

 <value name=*"processFlag"*>-1</value>

 <value name=*"processMessage"*>itxSvc:异常信息</value>

</Values>

或者返回html

***解决办法：***如果收到上述返回报文都是因为dom4j解析报文失败，可以网上搜索content is not allowed in prolog的异常产生原因，常见原因为非法字符，或者用note等文本编译器保存致UTF-8编码时候，会在前面加0xFE之类的BOM，导致一些xml解析不识别，请放到txt中保存。