普兰泰科报价引擎客户端使用说明书

V1.0

╳╳╳╳-╳╳-╳╳实施

╳╳╳╳-╳╳-╳╳发布

目 录

[客户使用接口类图 3](#_Toc395887119)

[1 IClientConneted 3](#_Toc395887120)

[2 IBussiness 4](#_Toc395887121)

# 客户使用接口类图



# IClientConneted

* 1. 接口说明

该接口主要提供方法供用户调用.主要实现用户向报价引擎发送各种请求.例如登录请求,订阅请求.

* 1. 接口方法

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **方法名称** | **返回值** | **方法说明** | **参数** | **参数说明** |
| conneteLocateGateWay | void | 向报价引擎登录注册客户信息. | serverAddress: String,  port: int,  userName:String,  password: String | 报价引擎地址  报价引擎端口  用户名  用户密码 |
| openRICMarket | void | 向报价引擎发起订阅产品请求 | RIC: String | RIC编码 |

* 1. 示例

IClientConneted clientConnetor = new ClientConnector(bussinessHandler);

//向服务器注册客户端信息.参数1服务器ip,参数2服务器端口,参数3客户名称,参数4客户密码.

sample.clientConnetor.conneteLocateGateWay("61.144.244.173", 8888, "ztcj", "ztcj2013");

//向服务器发送RIC请求.BussinessHandler的handleMessage方法就可以接收到服务器返回的市场价格了.

sample.clientConnetor.openRICMarket("XAU=");

sample.clientConnetor.openRICMarket("XAG=");

# IBussiness

* 1. 接口说明

该接口主要提供报价引擎报价接收,服务端异常处理.

该接口所有方法需要用户自行实现.

* 1. 接口方法

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **方法名称** | **返回值** | **方法说明** | **参数** | **参数说明** |
| handleMessage | void | 处理报价引擎发送过来的报价信息. | message: String | 报价引擎发送的报价,格式一般为JSON或者XML. |
| handleException | void | 处理通讯异常 | e: Throwable | 异常具体信息 |
| handleDisconnected | Void | 处理网络连接断开 |  |  |

* 1. 示例

public class RFASample {

//向服务器发送请求的客户端接口.

public IClientConnected clientConnetor;

//实例化

public RFASample(){

//BussinessInterface该接口是接受服务器数据的接口.这个接口请客户一定要自己实现.

IBussiness bussinessHandler = new BussinessHandler();

//ClientConnectedInterface向服务器发送请求的接口.这个接口客户无需实现.只要调用这个ClientConnector这个类里面的方法就好了.

clientConnetor = new ClientConnector(bussinessHandler);

}

class BussinessHandler implements IBussiness{

/\*

\* 客户自行实现实现处理任何网络异常的代码.

\*/

@Override

public void handleException(Throwable e){

System.out.println(e);

}

/\*

\* 客户实现数据处理的代码.参数message是服务器发送过来的字符串形式的数据.

\* 可以用Dom4j直接转换成Dom形式.

\*/

@Override

public void handleMessage(String message){

Document document = XmlMessageUtil.convertDocument(message);

byte msgType = XmlMessageUtil.getMsgType(document);

if (document == null) {

System.out.println("Received server's message is null \n");

return;

}

switch(msgType){

//首先服务器会发送过来一个snapshot的信息.里面包括该RIC对应的所有字段.

case MsgType.REFRESH\_RESP:

System.out.println(document);

break;

//然后服务器会发送很多更新的MarketPrice.该信息只包括需要更新的字段.

//如RIC: XAU= 只会发送BID,BID1,BID2,ASK,ASK1,ASK2等字段.

case MsgType.UPDATE\_RESP:

System.out.println(document);

break;

//如果服务器有通知服务器状态改变的信息,会使用此状态信息.

case GateWayMessageTypes.RESPONSE\_LOGIN:

case MsgType.STATUS\_RESP:

System.out.println("");

break;

//服务器发送了未知的消息,一般这里不用处理.扔掉该消息就好了.

default:

System.out.println("Not should to here! message type is "+MsgType.REFRESH\_RESP);

}

// String content = response.asXML();

System.out.println("Received server's message : " + message+"\n");

}

@Override

public void handleDisconnected() {

System.out.println("Locate Server disconnted!!! ");

}

}

public static void main(String[] args) {

RFASample sample = new RFASample();

//向服务器注册客户端信息.参数1服务器ip,参数2服务器端口,参数3客户名称,参数4客户密码.

sample.clientConnetor.conneteLocateGateWay("61.144.244.173", 8888, "ztcj", "ztcj2013");

//向服务器发送RIC请求.BussinessHandler的handleMessage方法就可以接收到服务器返回的市场价格了.

sample.clientConnetor.openRICMarket("XAU=");

sample.clientConnetor.openRICMarket("XAG=");

}

}