融航交易平台

风

控

应

用

程

序

接

口

2019 年 5 月

1.文件属性

文件属性	内容
文件名称	融航交易平台风控 API 接口
文件编号	
文件版本号	V1.1
文件状态	发布
作者	上海融航信息技术股份有限公司
文档编写日期	2018-9-14
文档发布日期	2019-5-13

2.文件变更历史清单

文件版本号	修正日期	修正人	备 注
1.0	2018-9-14	交易服务开发组	创建文档
1.1	2019-5-13	交易服务开发组	增加接口

3.本次修改变更说明

序号	变更内容简述
1.	增加订阅回报接口，增加报单和成交主推接口，增加查询资金接口
2.	
3.	
4.	
5.	
6.	

7.	
8.	
9.	
10.	
11.	
12.	
13.	

目录

1.介绍

融航交易平台风控 API 是一个基于 C++的类库, 通过使用和扩展类库提供的接口来实现登入、登出、查询风控账户关联交易账户、查询关联交易账户持仓等功能。

文件名	版本	文件大小	文件描述
RHUserApiStruct.h	V1.0	13755 字节	定义了 API 所需的一系列数据类型的头文件
RHUserApiDataType.h	V1.0	6,803 字节	定义了一系列业务相关的数据结构的头文件
RHMonitorApi.h	V1.0	2,839 字节	风控接口头文件

支持 MS VC 6.0 ， MS VC.NET 2010 编译器 。 需要打开多线程编译选项/MT。

2.体系结构

融航交易平台风控 API 使用建立在 TCP 协议之上风控特有协议与融航交易平台进行通讯，融航交易平台负责投资者的交易业务处理和风控参指标的计算。

3.接口模式

融航交易平台风控 API 提供了二个接口 ，分别为 CRHMonitorApi 和 CRHMonitorSpi。这两个接口对融航风控数据协议进行了封装 ，方便客户端应用程序的开发。客户端应用程序可以通过 CRHMonitorApi 发出操作请求，通过继承 CRHMonitorSpi 并重载回调函数来处理后台服务的响应。

3.1.对话流和查询流编程接口

通过对话流进行通讯的编程接口通常如下

请求：

```
int CRHMonitorApi::ReqXXX(
    CRHXXXField *pReqXXX,
    int nRequestID)
```

响应：

```
void CRHMonitorSpi::OnRspXXX(
    CRHRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast)
```

其中请求接口第一个参数为请求的内容，不能为空。

第二个参数为请求号 。 请求号由客户端应用程序负责维护 ， 正常情况下每个请求的请求号不要重复 。在接收融航交易平台的响应时 ，可以得到当时发出请求时填写的请求号，从而可以将响应与请求对应起来。

当收到后台服务应答时 ， CRHMonitorSpi 的回调函数会被调用。如果响应数据不止一个，则回调函数会被多次调用。

回调函数的第一个参数为响应的具体数据，如果出错或没有结果有可能为 NULL。 第二个参数为处理结果 ，表明本次请求的处理结果是成功还是失败 。 在发生多次回调时，除了第一次回调，其它的回调该参数都可能为 NULL。第三个参数为请求号，即原来发出请求时填写的请求号。 第四个参数为响应结束标志，表明是否是本次响应的最后一次回调。

4.开发接口

4.1.通用规则

客户端和融航交易平台的通讯过程分为 2 个阶段：初始化阶段和功能调用阶段。在初始化阶段，程序必须完成如下步骤（具体代码请参考开发实例 ） ：

1, 产生一个 CRHMonitorApi 实例

2, 产生一个事件处理的实例

3, 注册一个事件处理的实例

4, 设置交易托管服务的地址

5, 在功能调用阶段，程序可以任意调用风控接口中的请求方法，如 ReqQryInvestorPosition 等。同时按照需要响应回调接口中的。

其他注意事项：

1，API 请求的输入参数不能为 NULL。

2，API 请求的返回参数，0 表示正确，其他表示错误，详细错误编码请查表。

4.2. CRHMonitorSpi 接口

CRHMonitorSpi 实现了事件通知接口。用户必需派生 CRHMonitorSpi 接口，编写事件处理方法来处理感兴趣的事件。

4.2.1.OnFrontConnected 方法

当客户端与融航交易平台建立起通信连接时（还未登录前），该方法被调用。

函数原形：

void OnFrontConnected()；

本方法在完成初始化后调用，可以在其中完成用户登录任务。

4.2.2.OnFrontDisconnected 方法

当客户端与融航交易平台通信连接断开时，该方法被调用。当发生这个情况后，API 会自动重新连接，客户端可不做处理。

函数原形：

void OnFrontDisconnected (int nReason)；

参数：

nReason：连接断开原因

0x1001 网络读失败

0x1002 网络写失败

0x2001 接收心跳超时

0x2002 发送心跳失败

0x2003 收到错误报文

4.2.3.OnRspUserLogin 方法

当客户端发出登录请求之后，融航交易平台返回响应时，该方法会被调用，通知客户端登录是否成功。

函数原形：

void OnRspUserLogin(

CRHMonitorRspUserLoginField*pRspUserLoginField,

CRHRspInfoField*pRHRspInfoField,

int nRequestID,

bool bIsLast)；

参数： pRspUserLoginField：返回用户登录信息的地址。 用户登录信息结构：

```
struct CRHMonitorRspUserLoginField
{
    //风控账号
    TRHUserIDType           UserID;
    //账户权限
    TRHPrivilegeType        PrivilegeType;
    //信息查看权限
    TRHInfoPrivilegeType    InfoPrivilegeType;
    ///交易日
    TRHDateType             TradingDay;
    ///登录成功时间
    TRHTimeType             LoginTime;
};
```

pRHRspInfoField： 返回用户响应信息的地址 。 特别注意在有连续的成功的响应数据时，中间有可能返回 NULL，但第一次不会，以下同。错误代码为 0 时，表示操作成功，以下同。

```
///响应信息
struct CRHRspInfoField
{
    ///错误代码
    TRHErrorIDType      ErrorID;
    ///错误信息
    TRHErrorMsgType   ErrorMsg;
};
```

nRequestID：返回用户登录请求的 ID，该 ID 由用户在登录时指定。

bIsLast：指示该次返回是否为针对 nRequestID 的最后一次返回。

4.2.4.OnRspUseLogout 方法

当客户端发出退出请求之后，融航交易平台返回响应时，该方法会被调用，通知客户端退出是否成功。

函数原形：

```
void OnRspUserLogout(
CRHMonitorUserLogoutField * pRspUserLoginField,
CRHRspInfoField * pRHRspInfoField,
int nRequestID)；
```

参数：

pRspUserLoginField：返回用户退出信息的地址。

用户登出信息结构：

```
struct CRHMonitorUserLogoutField
{
    //风控账号
    TRHUserIDType            UserID;
};
```

pRHRspInfoField：返回用户响应信息的地址。

响应信息结构：

```
struct CRHRspInfoField
{
    ///错误代码
    TRHErrorIDType    ErrorID;
    ///错误信息
    TRHErrorMsgType   ErrorMsg;
};
```

nRequestID：返回用户登出请求的 ID，该 ID 由用户在登出时指定。

bIsLast：指示该次返回是否为针对 nRequestID 的最后一次返回。

4.2.5. OnRspQryInvestorPosition 方法

风控关联交易账户持仓查询应答。当客户端发出该查询指令后，融航交易平台返回响应时，方法会被调用。

函数原形：

```
void OnRspQryInvestorPosition (
CRHMonitorPositionField* pRHMonitorPositionField,
CRHRspInfoField* pRHRspInfoField,
int nRequestID,
bool bIsLast)；
```

参数：

pRHMonitorPositionField：指向交易账户持仓信息结构的地址。

交易账户持仓信息结构：

```
//持仓监控信息
struct CRHMonitorPositionField
{
    ///投资者代码
    TRHInvestorIDType InvestorID;
    ///经纪公司代码
    TRHBrokerIDType        BrokerID;
    ///品种类别
    TRHInstrumentIDType    ProductID;
    ///合约代码
    TRHInstrumentIDType    InstrumentID;
    ///投机套保标志
    TRHHedgeFlagType       HedgeFlag;
    ///持仓方向
    TRHDirectionType       Direction;
```

```
    ///持仓数量
    TRHVolumeType          Volume;
    ///持仓保证金
    TRHMoneyType               Margin;
    ///逐笔开仓均价
    TRHMoneyType               AvgOpenPriceByVol;
    ///逐日开仓均价
    TRHMoneyType               AvgOpenPrice;
    ///今仓数量
    TRHVolumeType          TodayVolume;
    ///冻结持仓数量
    TRHVolumeType          FrozenVolume;
    ///信息类型
    TRHPositionEntryType   EntryType;
    ///昨仓，冻结持仓数量，逐笔持盈，逐笔开仓均价
};
```

pRHMonitorPositionField：指向响应信息结构的地址。

响应信息结构：

```
struct CRHRspInfoField
{
    ///错误代码
    TRHErrorIDType     ErrorID;
    ///错误信息
    TRHErrorMsgType   ErrorMsg;
};
```

nRequestID：返回会员客户查询请求的 ID，该 ID 由用户在会员客户查询时指定。

bIsLast 指示该次返回是否为针对 nRequestID 的最后一次返回。

4.2.6. OnRspQryMonitorAccounts 方法

请求查询当前风控账户关联监控的交易账户信息响应。当客户端发出请求查询资金账户指令后，交易托管系统返回响应时，该方法会被调用。

函数原形：

```
void OnRspQryMonitorAccounts (
CRHQryInvestorField* pRspMonitorUser,
CRHRspInfoField* pRHRspInfoField,
int nRequestID,
bool bIsLast)；
```

参数：

pRspMonitorUser：指向风控账户关联的交易账户结构的地址。

```
///查询投资者
struct CRHQryInvestorField
{
    ///经纪公司代码
    TRHBrokerIDType    BrokerID;
    ///投资者代码
    TRHInvestorIDType  InvestorID;
};
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CRHRspInfoField
{
    ///错误代码
    TRHErrorIDType     ErrorID;
    ///错误信息
    TRHErrorMsgType    ErrorMsg;
};
```

nRequestID：返回会员客户查询请求 ID，该 ID 由用户在会员客户查询时指定。

bIsLast：指示该次返回是否为针对 nRequestID 的最后一次返回。

4.2.7. OnRspQryInvestorMoney 方法

请求查询指定账户资金信息响应。当客户端发出请求查询账户资金指令后，交易托管系统返回响应时，该方法会被调用。

函数原形：

void OnRspQryInvestorMoney(CRHTradingAccountField* pRHTradingAccountField, CRHRspInfoField* pRHRspInfoField, int nRequestID,bool isLast);

参数：

pRHTradingAccountField：指向账户资金信息结构的地址。

```
///资金账户
struct CRHTradingAccountField
{
    ///经纪公司代码
    TRHBrokerIDType    BrokerID;
    ///投资者帐号
    TRHAccountIDType AccountID;
    ///上次质押金额
    TRHMoneyType      PreMortgage;
    ///上次信用额度
    TRHMoneyType      PreCredit;
    ///上次存款额
    TRHMoneyType      PreDeposit;
    ///上次结算准备金
    TRHMoneyType      PreBalance;
    ///上次占用的保证金
```

```
TRHMoneyType	PreMargin;
///利息基数
TRHMoneyType	InterestBase;
///利息收入
TRHMoneyType	Interest;
///入金金额
TRHMoneyType	Deposit;
///出金金额
TRHMoneyType	Withdraw;
///冻结的保证金
TRHMoneyType	FrozenMargin;
///冻结的资金
TRHMoneyType	FrozenCash;
///冻结的手续费
TRHMoneyType	FrozenCommission;
///当前保证金总额
TRHMoneyType	CurrMargin;
///资金差额
TRHMoneyType	CashIn;
///手续费
TRHMoneyType	Commission;
///平仓盈亏
TRHMoneyType	CloseProfit;
///持仓盈亏
TRHMoneyType	PositionProfit;
///期货结算准备金
TRHMoneyType	Balance;
///可用资金
TRHMoneyType	Available;
///可取资金
```

```
TRHMoneyType	WithdrawQuota;
///基本准备金
TRHMoneyType	Reserve;
///交易日
TRHDateType	TradingDay;
///结算编号
TRHSettlementIDType	SettlementID;
///信用额度
TRHMoneyType	Credit;
///质押金额
TRHMoneyType	Mortgage;
///交易所保证金
TRHMoneyType	ExchangeMargin;
///投资者交割保证金
TRHMoneyType	DeliveryMargin;
///交易所交割保证金
TRHMoneyType	ExchangeDeliveryMargin;
///保底期货结算准备金
TRHMoneyType	ReserveBalance;
///币种代码
TRHCurrencyIDType	CurrencyID;
///上次货币质入金额
TRHMoneyType	PreFundMortgageIn;
///上次货币质出金额
TRHMoneyType	PreFundMortgageOut;
///货币质入金额
TRHMoneyType	FundMortgageIn;
///货币质出金额
TRHMoneyType	FundMortgageOut;
///货币质押余额
```

```
TRHMoneyType	FundMortgageAvailable;
///可质押货币金额
TRHMoneyType	MortgageableFund;
///特殊产品占用保证金
TRHMoneyType	SpecProductMargin;
///特殊产品冻结保证金
TRHMoneyType	SpecProductFrozenMargin;
///特殊产品手续费
TRHMoneyType	SpecProductCommission;
///特殊产品冻结手续费
TRHMoneyType	SpecProductFrozenCommission;
///特殊产品持仓盈亏
TRHMoneyType	SpecProductPositionProfit;
///特殊产品平仓盈亏
TRHMoneyType	SpecProductCloseProfit;
///根据持仓盈亏算法计算的特殊产品持仓盈亏
TRHMoneyType	SpecProductPositionProfitByAlg;
///特殊产品交易所保证金
TRHMoneyType	SpecProductExchangeMargin;
///业务类型
TRHBizTypeType	BizType;
///延时换汇冻结金额
TRHMoneyType	FrozenSwap;
///剩余换汇额度
TRHMoneyType	RemainSwap;
//证券持仓市值
TRHMoneyType TotalStockMarketValue;
//期权持仓市值
TRHMoneyType TotalOptionMarketValue;
//动态权益
```

```
    TRHMoneyType DynamicMoney;
    ///权利金收支
    TRHMoneyType Premium;
    ///市值权益
    TRHMoneyType MarketValueEquity;
};
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CRHRspInfoField
{
    ///错误代码
    TRHErrorIDType      ErrorID;
    ///错误信息
    TRHErrorMsgType   ErrorMsg;
};
```

nRequestID：返回会员客户查询请求 ID，该 ID 由用户在会员客户查询时指定。

bIsLast：指示该次返回是否为针对 nRequestID 的最后一次返回。

4.2.8. OnRspOffsetOrder 方法

API 发起合约强平指令失败响应。当 API 发出合约强平指令后，如果交易系统判断该强平无效，交易托管系统返回响应时，该方法会被调用。

函数原形：

```
void OnRspOffsetOrder(CRHMonitorOffsetOrderField*
pMonitorOrderField,CRHRspInfoField* pRHRspInfoField, int nRequestID,bool isLast);
```

参数：

pMonitorOrderField：指向强平指令委托时信息结构的地址。

```
//风控端强制平仓字段
struct CRHMonitorOffsetOrderField
```

```
{
    //投资者
    TRHInvestorIDType    InvestorID;
    //经纪公司代码
    TRHBrokerIDType          BrokerID;
    //合约 ID
    TRHInstrumentIDType      InstrumentID;
    //方向
    TRHDirectionType         Direction;
    //手数
    TRHVolumeType            volume;
    //价格
    TRHPriceType             Price;
    ///组合开平标志
    TRHCombOffsetFlagType    CombOffsetFlag;
    ///组合投机套保标志
    TRHCombHedgeFlagType         CombHedgeFlag;
};
```

pRspInfo：指向响应信息结构的地址。

响应信息结构：

```
struct CRHRspInfoField
{
    ///错误代码
    TRHErrorIDType    ErrorID;
    ///错误信息
    TRHErrorMsgType   ErrorMsg;
};
```

nRequestID：返回会员客户查询请求 ID，该 ID 由用户在会员客户查询时指定。

bIsLast：指示该次返回是否为针对 nRequestID 的最后一次返回。

4.2.9. OnRtnOrder 方法

在 API 订阅某账户的委托回报后，如果该账户有新的委托回报产生，该方法会被调用。

函数原形：

```
void OnRtnOrder(CRHOrderField *pOrder);
```

参数：

pOrder：指向报单回报信息结构的地址。

```
///报单
struct CRHOrderField
{
    ///经纪公司代码
    TRHBrokerIDType    BrokerID;
    ///投资者代码
    TRHInvestorIDType  InvestorID;
    ///合约代码
    TRHInstrumentIDType    InstrumentID;
    ///报单引用
    TRHOrderRefType    OrderRef;
    ///用户代码
    TRHUserIDType      UserID;
    ///报单价格条件
    TRHOrderPriceTypeType OrderPriceType;
    ///买卖方向
    TRHDirectionType   Direction;
    ///组合开平标志
    TRHCombOffsetFlagType      CombOffsetFlag;
    ///组合投机套保标志
    TRHCombHedgeFlagType       CombHedgeFlag;
    ///价格
    TRHPriceType  LimitPrice;
```

```
///数量
TRHVolumeType     VolumeTotalOriginal;
///有效期类型
TRHTimeConditionType  TimeCondition;
///GTD 日期
TRHDateType  GTDDate;
///成交量类型
TRHVolumeConditionType     VolumeCondition;
///最小成交量
TRHVolumeType     MinVolume;
///触发条件
TRHContingentConditionType       ContingentCondition;
///止损价
TRHPriceType  StopPrice;
///强平原因
TRHForceCloseReasonType    ForceCloseReason;
///自动挂起标志
TRHBoolType  IsAutoSuspend;
///业务单元
TRHBusinessUnitType     BusinessUnit;
///请求编号
TRHRequestIDType RequestID;
///本地报单编号
TRHOrderLocalIDType     OrderLocalID;
///交易所代码
TRHExchangeIDType       ExchangeID;
///会员代码
TRHParticipantIDType     ParticipantID;
///客户代码
TRHClientIDType     ClientID;
```

///合约在交易所的代码

TRHExchangeInstIDType ExchangeInstID;

///交易所交易员代码

TRHTraderIDType TraderID;

///安装编号

TRHInstallIDType InstallID;

///报单提交状态

TRHOrderSubmitStatusType OrderSubmitStatus;

///报单提示序号

TRHSequenceNoType NotifySequence;

///交易日

TRHDateType TradingDay;

///结算编号

TRHSettlementIDType SettlementID;

///报单编号

TRHOrderSysIDType OrderSysID;

///报单来源

TRHOrderSourceType OrderSource;

///报单状态

TRHOrderStatusType OrderStatus;

///报单类型

TRHOrderTypeType OrderType;

///今成交数量

TRHVolumeType VolumeTraded;

///剩余数量

TRHVolumeType VolumeTotal;

///报单日期

TRHDateType InsertDate;

///委托时间

TRHTimeType InsertTime;

///激活时间

TRHTimeType ActiveTime;

///挂起时间

TRHTimeType SuspendTime;

///最后修改时间

TRHTimeType UpdateTime;

///撤销时间

TRHTimeType CancelTime;

///最后修改交易所交易员代码

TRHTraderIDType ActiveTraderID;

///结算会员编号

TRHParticipantIDType ClearingPartID;

///序号

TRHSequenceNoType SequenceNo;

///前置编号

TRHFrontIDType FrontID;

///会话编号

TRHSessionIDType SessionID;

///用户端产品信息

TRHProductInfoType UserProductInfo;

///状态信息

TRHErrorMsgType StatusMsg;

///用户强评标志

TRHBoolType UserForceClose;

///操作用户代码

TRHUserIDType ActiveUserID;

///经纪公司报单编号

TRHSequenceNoType BrokerOrderSeq;

///相关报单

TRHOrderSysIDType RelativeOrderSysID;

```
    ///郑商所成交数量
    TRHVolumeType     ZCETotalTradedVolume;
    ///互换单标志
    TRHBoolType   IsSwapOrder;
    ///营业部编号
    TRHBranchIDType    BranchID;
    ///投资单元代码
    TRHInvestUnitIDType     InvestUnitID;
    ///资金账号
    TRHAccountIDType AccountID;
    ///币种代码
    TRHCurrencyIDType       CurrencyID;
    ///IP 地址
    TRHIPAddressType  IPAddress;
    ///Mac 地址
    TRHMacAddressType     MacAddress;
};
```

4.2.10. OnRtnTrade 方法

在 API 订阅某账户的成交回报后，如果该账户有新的成交信息产生，该方法会被调用。

函数原形：

```
void OnRtnTrade(CRHTradeField *pTrade);
```

参数：

```
///成交
struct CRHTradeField
{
    ///经纪公司代码
    TRHBrokerIDType    BrokerID;
    ///投资者代码
```

TRHInvestorIDType InvestorID;

///合约代码

TRHInstrumentIDType InstrumentID;

///报单引用

TRHOrderRefType OrderRef;

///用户代码

TRHUserIDType UserID;

///交易所代码

TRHExchangeIDType ExchangeID;

///成交编号

TRHTradeIDType TradeID;

///买卖方向

TRHDirectionType Direction;

///报单编号

TRHOrderSysIDType OrderSysID;

///会员代码

TRHParticipantIDType ParticipantID;

///客户代码

TRHClientIDType ClientID;

///交易角色

TRHTradingRoleType TradingRole;

///合约在交易所的代码

TRHExchangeInstIDType ExchangeInstID;

///开平标志

TRHOffsetFlagType OffsetFlag;

///投机套保标志

TRHHedgeFlagType HedgeFlag;

///价格

TRHPriceType Price;

///数量

```
    TRHVolumeType     Volume;
    ///成交时期
    TRHDateType  TradeDate;
    ///成交时间
    TRHTimeType  TradeTime;
    ///成交类型
    TRHTradeTypeType TradeType;
    ///成交价来源
    TRHPriceSourceType      PriceSource;
    ///交易所交易员代码
    TRHTraderIDType    TraderID;
    ///本地报单编号
    TRHOrderLocalIDType    OrderLocalID;
    ///结算会员编号
    TRHParticipantIDType    ClearingPartID;
    ///业务单元
    TRHBusinessUnitType    BusinessUnit;
    ///序号
    TRHSequenceNoType    SequenceNo;
    ///交易日
    TRHDateType  TradingDay;
    ///结算编号
    TRHSettlementIDType    SettlementID;
    ///经纪公司报单编号
    TRHSequenceNoType    BrokerOrderSeq;
    ///成交来源
    TRHTradeSourceType    TradeSource;
    ///投资单元代码
    TRHInvestUnitIDType    InvestUnitID;
};
```

4.3. CRHMonitorApi 接口

CRHMonitorApi 接口提供给用户的功能包括，登入、登出、查询关联的交易账户、查询交易账户的持仓信息等功能。

4.3.1. CreateRHMonitorApi 方法

产生一个 CRHMonitorApi 的一个实例，不能通过 new 来产生。

函数原形：

static CRHMonitorApi *CreateRHMonitorApi();

参数：无

返回值：

返回一个指向 CRHMonitorApi 实例的指针。

4.3.2.Release 方法

释放一个 CRHMonitorApi 实例。不能使用 delete 方法

函数原形：

void Release()；

4.3.3.Init 方法

使客户端开始与融航交易平台建立连接，连接成功后可以进行登陆。

函数原形：

void Init(const char * ip,unsigned int port)；

参数：

Ip: 即融航交易服务器的 IP 地址

Port: 即融航交易服务器的端口

4.3.4.RegisterSpi 方法

注册一个派生自 CRHMonitorSpi 接口类的实例，该实例将完成事件处理。

函数原形：

void RegisterSpi(CRHMonitorSpi*pSpi) ;

参数：

pSpi 实现了 CRHMonitorSpi 接口的实例指针。

4.3.5.ReqUserLogin 方法

用户发出登陆请求。

函数原形：

int ReqUserLogin(

CRHMonitorReqUserLoginField*pUserLoginField,

Int nRequestID)；

参数：

pUserLoginField：指向用户登录请求结构的地址。

用户登录请求结构：

```
struct CRHMonitorReqUserLoginField
{
    //风控账号
    TRHUserIDType           UserID;
    //风控密码
    TRHPasswordType         Password;
```

```
    //MAC 地址
    TRHMacAddressType        MacAddress;
};
```

nRequestID：用户登录请求的 ID，该 ID 由用户指定，管理。

用户需要填写 MacAddress 字段，即客户端的 MAC 地址信息，如果不填写，API 将会自动获取用户机器的 MAC 信息。

返回值：

0，代表成功。

-1，表示网络连接失败；

4.3.6.ReqUserLogout 方法

用户发出登出请求。

函数原形：

```
int ReqUserLogout(
CRHMonitorUserLogoutField * pUserLogoutField,
int nRequestID)；
```

参数：

pUserLogoutField：指向用户登出请求结构的地址。

用户登出请求结构：

```
struct CRHMonitorUserLogoutField
{
    //风控账号
    TRHUserIDType              UserID;
};
```

nRequestID：用户登出请求的 ID，该 ID 由用户指定，管理。

返回值：

0,代表成功。

-1，表示网络连接失败；

4.3.7.ReqQryInvestorPosition 方法

交易账户持仓查询请求。

函数原形：

```
int ReqQryInvestorPosition (
CRHMonitorQryInvestorPositionField * pQryInvestorPositionField,
int nRequestID)；
```

参数：

pQryInvestorPositionField：指向交易账户持仓查询结构的地址。

会员持仓查询结构：

```
struct CRHMonitorQryInvestorPositionField
{
    ///投资者代码
    TRHInvestorIDType InvestorID;
    ///账户类别
    //TRHAccountType      AccountType;
    ///经纪公司代码
    TRHBrokerIDType       BrokerID;
    ///合约代码
    TRHInstrumentIDType   InstrumentID;
};
```

nRequestID：会员持仓查询请求的 ID，该 ID 由用户指定，管理。

返回值：

0，代表成功。

-1，表示网络连接失败；

-2，表示未处理请求超过许可数；

-3，表示每秒发送请求数超过许可数。

4.3.8. ReqQryMonitorAccounts 方法

请求查询风控账户关联的交易账户。

函数原形：

```
int ReqQryMonitorAccounts (
CRHMonitorQryMonitorUser * pQryMonitorUser,
int nRequestID)；
```

参数：

pQryMonitorUser：指向查询风控关联交易账户结构的地址。

```
struct CRHMonitorQryMonitorUser
{
    //风控账号
    TRHUserIDType              UserID;
};
```

nRequestID：会员持仓查询请求的 ID，该 ID 由用户指定，管理。

返回值：

0，代表成功。

-1，表示网络连接失败；

-2，表示未处理请求超过许可数；

-3，表示每秒发送请求数超过许可数。

4.3.9. ReqQryInvestorMoney 方法

交易账户资金查询请求。

函数原形：

```
Int          RHMoniterClient::ReqQryInvestorMoney(CRHMonitorQryInvestorMoneyField*
pQryInvestorMoneyField, int nRequestID)
```

参数：

pQryInvestorMoneyField：指向交易账户资金查询结构的地址。

账户资金查询结构：

```
struct CRHMonitorQryInvestorMoneyField
{
    ///投资者代码
    TRHInvestorIDType InvestorID;
    ///经纪公司代码
    TRHBrokerIDType        BrokerID;    //该字段为空

};
```

nRequestID：账户资金查询请求的 ID，该 ID 由用户指定，管理。

返回值：

0，代表成功。

-1，表示网络连接失败；

-2，表示未处理请求超过许可数；

-3，表示每秒发送请求数超过许可数。

4.3.10. ReqOffsetOrder 方法

发起强平报单指令请求

函数原形：

```
int RHMoniterClient::ReqOffsetOrder(CRHMonitorOffsetOrderField* pMonitorOrderField, int
nRequestID)
```

参数：

pMonitorOrderField：指向强平指令参数结构的地址。

强平指令参数结构：

```
//强制平仓字段
struct CRHMonitorOffsetOrderField
```

```
{
    //投资者
    TRHInvestorIDType    InvestorID;
    //经纪公司代码
    TRHBrokerIDType          BrokenID;
    //合约 ID
    TRHInstrumentIDType      InstrumentID;
    //方向
    TRHDirectionType         Direction;
    //手数
    TRHVolumeType            volume;
    //价格
    TRHPriceType             Price;
    ///组合开平标志
    TRHCombOffsetFlagType    CombOffsetFlag;
    ///组合投机套保标志
    TRHCombHedgeFlagType         CombHedgeFlag;
};
```

nRequestID：强平指令请求的 ID，该 ID 由用户指定，管理。

返回值：

0，代表成功。

-1，表示网络连接失败；

-2，表示未处理请求超过许可数；

-3，表示每秒发送请求数超过许可数。

4.3.11. ReqSubPushInfo 方法

订阅主动推送信息

int ReqSubPushInfo(CRHMonitorSubPushInfo *pInfo, int nRequestID)

参数：

pInfo：指向订阅类型参数结构的地址。

订阅参数结构：

```
//订阅推送信息
struct CRHMonitorSubPushInfo
{
    ///投资者代码
    TRHInvestorIDType InvestorID;
    ///账户类别
    TRHAccountType      AccountType;
    ///经纪公司代码
    TRHBrokerIDType     BrokerID;
    ///订阅类型
    RHMonitorSubPushInfoType SubInfoType;
};
```

nRequestID：订阅请求的 ID，该 ID 由用户指定，管理。

返回值：

0，代表成功。

-1，表示网络连接失败；

-2，表示未处理请求超过许可数；

-3，表示每秒发送请求数超过许可数。