

# 飞马平台 TraderAPI 说明书



## 目录

1. 介绍 .....	1
2. 体系结构 .....	2
2.1 通讯模式 .....	2
2.2 数据流 .....	3
3. 接口模式 .....	4
3.1 对话流和查询流编程接口 .....	4
3.2 私有流编程接口 .....	4
4. 运行模式 .....	6
4.1 工作线程 .....	6
4.2 本地文件 .....	6
5. 业务和接口对照 .....	7
6. 开发接口 .....	10
6.1 通用规则 .....	10
6.2 CUsTpFtdcTraderSpi 接口 .....	10
6.2.1 OnFrontConnected 方法 .....	10
6.2.2 OnFrontDisconnected 方法 .....	10
6.2.3 OnHeartBeatWarning 方法 .....	11
6.2.4 OnRspUserLogin 方法 .....	11
6.2.5 OnRspUserLogout 方法 .....	12
6.2.6 OnRspUserPasswordUpdate 方法 .....	13
6.2.7 OnRspError 方法 .....	14
6.2.8 OnRspOrderInsert 方法 .....	14
6.2.9 OnRspOrderAction 方法 .....	16
6.2.10 OnRspQryOrder 方法 .....	17
6.2.11 OnRspQryTrade 方法 .....	19
6.2.12 OnRspQryInvestorAccount 方法 .....	21
6.2.13 OnRspQryTradingCode 方法 .....	23
6.2.14 OnRspQryExchange 方法 .....	24

6.2.15	OnRspQryInstrument 方法.....	25
6.2.16	OnRspQryUserInvestor 方法.....	27
6.2.17	OnRspQryInvestorPosition 方法.....	28
6.2.18	OnRspQryComplianceParam 方法.....	29
6.2.19	OnRspQryInvestorFee 方法.....	30
6.2.20	OnRspQryInvestorMargin 方法.....	31
6.2.21	OnRspQuoteInsert 方法.....	33
6.2.22	OnRspQuoteAction 方法.....	34
6.2.23	OnRspForQuote 方法.....	35
6.2.24	OnRspMarginCombAction 方法.....	36
6.2.25	OnRspUserDeposit 方法.....	38
6.2.1	OnRspTransferMoney 方法 .....	39
6.2.2	OnRspQryInvestorLegPosition 方法 .....	40
6.2.3	OnRspQryInvestorCombPosition 方法 .....	41
6.2.4	OnRspQryExchangRate 方法 .....	42
6.2.5	OnRtnTrade 方法 .....	43
6.2.6	OnRtnOrder 方法 .....	45
6.2.7	OnRtnInstrumentStatus 方法 .....	47
6.2.8	OnRtnInvestorAccountDeposit 方法 .....	49
6.2.9	OnErrRtnOrderInsert 方法 .....	50
6.2.10	OnErrRtnOrderAction 方法.....	51
6.2.11	OnRtnQuote 方法.....	52
6.2.12	OnErrRtnQuoteInsert 方法.....	54
6.2.13	OnErrRtnQuoteAction 方法.....	55
6.2.14	OnRtnMarginCombinationLeg 方法.....	56
6.2.15	OnRtnMarginCombAction 方法.....	57
6.2.16	OnRtnUserDeposit 方法.....	58
6.3	CUSTPFtdcTraderApi 接口 .....	58
6.3.1	CreateFtdcTraderApi 方法 .....	58
6.3.2	Release 方法 .....	59

6.3.3	Init 方法 .....	59
6.3.4	Join 方法 .....	59
6.3.5	GetTradingDay 方法 .....	59
6.3.6	RegisterSpi 方法 .....	59
6.3.7	RegisterFront 方法 .....	60
6.3.8	RegisterNameServer 方法 .....	60
6.3.9	SubscribePrivateTopic 方法 .....	60
6.3.10	SubscribePublicTopic 方法 .....	61
6.3.11	SubscribeForQuote 方法 .....	61
6.3.12	ReqUserLogin 方法 .....	61
6.3.13	ReqUserLogout 方法 .....	62
6.3.14	ReqUserPasswordUpdate 方法 .....	63
6.3.15	ReqOrderInsert 方法 .....	63
6.3.16	ReqOrderAction 方法 .....	66
6.3.17	ReqUserDeposit 方法 .....	68
6.3.18	ReqTransferMoney 方法 .....	69
6.3.19	ReqQryOrder 方法 .....	70
6.3.20	ReqQryTrade 方法 .....	70
6.3.21	ReqQryInvestorAccount 方法 .....	71
6.3.22	ReqQryTradingCode 方法 .....	71
6.3.23	ReqQryExchange 方法 .....	72
6.3.24	ReqQryInstrument 方法 .....	72
6.3.25	ReqQryUserInvestor 方法 .....	73
6.3.26	ReqQryInvestorPosition 方法 .....	73
6.3.27	ReqQryComplianceParam 方法 .....	74
6.3.28	ReqQryInvestorFee 方法 .....	74
6.3.29	ReqQryInvestorMargin 方法 .....	75
6.3.30	ReqQuoteInsert 方法 .....	75
6.3.31	ReqQuoteAction 方法 .....	77
6.3.32	ReqForQuote 方法 .....	79

6.3.33	ReqMarginCombAction 方法.....	80
6.3.34	ReqQryInvestorCombPosition 方法.....	81
6.3.35	ReqQryInvestorLegPosition 方法.....	81
6.3.36	ReqQryExchangeRate 方法.....	82
7.	开发实例 .....	83

## 1. 介绍

飞马平台 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相关交易功能，包括报单录入、报单撤销、报单查询、成交单查询、投资者查询、投资者持仓查询、合约查询、交易日获取等。该类库包含以下 7 个文件：

文件名	版本	文件大小	文件描述
FtdcTraderApi.h			交易接口头文件
FtdcUserApiDataType.h			定义了 API 所需的一系列数据类型的头文件
FtdcUserApiStruct.h			定义了一系列业务相关的数据结构的头文件
USTPtraderapi.dll			动态链接库二进制文件
USTPtraderapi.lib			导入库文件
USTPmduserapi.dll			动态链接库二进制文件
USTPmduserapi.lib			导入库文件

支持 MS VC 6.0，MS VC.NET 2003 编译器。需要打开多线程编译选项/MT。

## 2. 体系结构

交易员 API 使用建立在 TCP 协议之上 FTD 协议与飞马平台进行通讯，飞马平台负责投资者的交易业务处理。

### 2.1 通讯模式

FTD 协议中的所有通讯都基于某个通讯模式。通讯模式实际上就是通讯双方协同工作的方式。

FTD 涉及的通讯模式共有三种：

- 对话通讯模式
- 私有通讯模式
- 广播通讯模式

对话通讯模式是指由用户端主动发起的通讯请求。该请求被交易所端接收和处理，并予以响应。例如报单、查询等。这种通讯模式与普通的客户/服务器模式相同。

私有通讯模式是指交易所端主动，向某个特定的用户发出的信息。例如成交回报等。

广播通讯模式是指交易所端主动，向市场中的所有用户都发出相同的信息。例如公告、市场公共信息等。

通讯模式和网络的连接不一定存在简单的一对一的关系。也就是说，一个网络连接中可能传送多种不同通讯模式的报文，一种通讯模式的报文也可以在多个不同的连接中传送。

无论哪种通讯模式，其通讯过程都如图 1 所示：

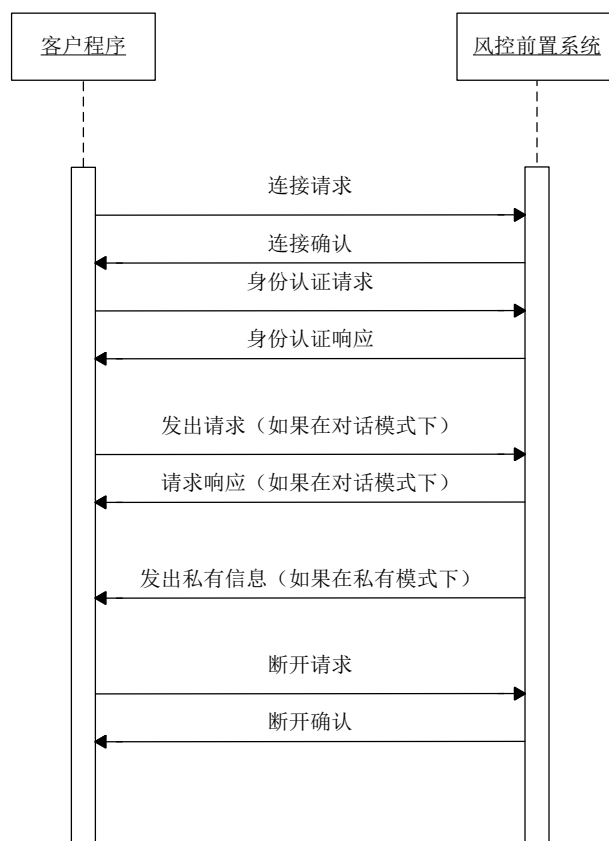


图 1-各种通讯模式的工作过程

本接口暂时没有使用广播通讯方式。

## 2.2 数据流

飞马平台支持对话通讯模式、私有通讯模式：

对话通讯模式下支持对话数据流和查询数据流：

对话数据流是一个双向数据流，飞马平台发送交易请求，交易系统反馈应答。交易系统不维护对话流的状态。系统故障时，对话数据流会重置，通讯途中的数据可能会丢失。

查询数据流是一个双向数据流，飞马平台发送查询请求，交易系统反馈应答。交易系统不维护查询流的状态。系统故障时，查询数据流会重置，通讯途中的数据可能会丢失。

私有通讯模式下支持私有数据流：

私有流是一个单向数据流，由交易系统发向飞马平台，用于传送交易员私有的通知和回报信息。私有流是一个可靠的数据流，交易系统维护每个飞马平台的私有流，在一个交易日内，飞马平台断线后恢复连接时，可以请求交易系统发送指定序号之后的私有流数据。私有数据流向飞马平台提供报单状态报告、成交回报等信息。



### 3. 接口模式

交易员 API 提供了二个接口,分别为 CUSTPFtdcTraderApi 和 CUSTPFtdcTraderSpi。这两个接口对 FTD 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 CUSTPFtdcTraderApi 发出操作请求,通继承 CUSTPFtdcTraderSpi 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 CUSTPFtdcTraderApi::ReqXXX(  
        CUSTPFtdcXXXField *pReqXXX,  
        int nRequestID)  
响应: void CUSTPFtdcTraderSpi::OnRspXXX(  
        CUSTPFtdcXXXField *pRspXXX,  
        CUSTPFtdcRspInfoField *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。在接收飞马平台的响应时,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,CUSTPFtdcTraderSpi 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。

#### 3.2 私有流编程接口

私有流中的数据中用户的私有信息,包括报单回报、成交回报等。

通过私有流接收回报的编程接口通常如下:

```
void CUSTPFtdcTraderSpi::OnRtnXXX(CUSTPFtdcXXXField *pXXX) 或
```

```
void CUSTPFtdcTraderSpi::OnErrRtnXXX(CUSTPFtdcXXXField *pXXX,  
    CUSTPFtdcRspInfoField *pRspInfo)
```

当收到飞马平台通过私有流发布的回报数据时，CUSTPFtdcTraderSpi 的回调函数会被调用。回调函数的参数为回报的具体内容。

## 4. 运行模式

### 4.1 工作线程

交易员客户端应用程序至少由两个线程组成，一个是应用程序主线程，一个是交易员 API 工作线程。应用程序与交易系统的通讯是由 API 工作线程驱动的。

CUSTPFtdcTraderApi 提供的接口是线程安全的，可以有多个应用程序线程同时发出请求。

CUSTPFtdcTraderSpi 提供的接口回调是由 API 工作线程驱动，通过实现 SPI 中的接口方法，可以从飞马平台收取所需数据。

如果重载的某个回调函数阻塞，则等于阻塞了 API 工作线程，API 与交易系统的通讯会停止。因此，在 CUSTPFtdcTraderSpi 派生类的回调函数中，通常应迅速返回，可以利用将数据放入缓冲区或通过 Windows 的消息机制来实现。

### 4.2 本地文件

交易员 API 在运行过程中，会将一些数据写入本地文件中。调用 CreateFtdcTraderApi 函数，可以传递一个参数，指明存贮本地文件的路径。该路径必须在运行前已创建好。本地文件的扩展名都是“.con”。

## 5. 业务和接口对照

业务类型	业务	请求接口	响应接口	数据流
登录	登录	CUSTPFtdcTraderApi::ReqUserLogin	CUSTPFtdcTraderSpi::OnRspUserLogin	对话流
	登出	CUSTPFtdcTraderApi::ReqUserLogout	CUSTPFtdcTraderSpi::OnRspUserLogout	对话流
	修改用户口令	CUSTPFtdcTraderApi::ReqUserPasswordUpdate	CUSTPFtdcTraderSpi::OnRspUserPasswordUpdate	对话流
报单	报单录入	CUSTPFtdcTraderApi::ReqOrderInsert	CUSTPFtdcTraderSpi::OnRspOrderInsert	对话流
	报单操作	CUSTPFtdcTraderApi::ReqOrderAction	CUSTPFtdcTraderSpi::OnRspOrderAction	对话流
报价	报价录入	CUSTPFtdcTraderApi::ReqQuoteInsert	CUSTPFtdcTraderSpi::OnRspQuoteInsert	对话流
	报价操作	CUSTPFtdcTraderApi::ReqQuoteAction	CUSTPFtdcTraderSpi::OnRspQuoteAction	对话流
询价	询价请求	CUSTPFtdcTraderApi::ReqForQuote	CUSTPFtdcTraderSpi::OnRspForQuote	对话流
组保	组合拆分请求	CUSTPFtdcTraderApi::ReqMarginCombAction	CUSTPFtdcTraderSpi::OnRspMarginCombAction	对话流
出入金	用户出入金请求	CUSTPFtdcTraderApi::ReqUserDeposit	CUSTPFtdcTraderSpi::OnRspUserDeposit	对话流
私有回报	成交回报	N/A	CUSTPFtdcTraderSpi::OnRtnTrade	私有流
	报单回报	N/A	CUSTPFtdcTraderSpi::OnRtnOrder	私有流

	柜台出入金回报	N/A	CUSTPFtdcTraderSpi::OnRtnInvestorAccountDeposit	私有流
	报单录入错误回报	N/A	CUSTPFtdcTraderSpi::OnErrRtnOrderInsert	私有流
	报单操作错误回报	N/A	CUSTPFtdcTraderSpi::OnErrRtnOrderAction	私有流
	报价回报	N/A	CUSTPFtdcTraderSpi::OnRtnQuote	私有流
	报价录入错误回报	N/A	CUSTPFtdcTraderSpi::OnErrRtnQuoteInsert	私有流
	用户出入金回报	N/A	CUSTPFtdcTraderSpi::OnRtnUserDeposit	私有流
组保回报	组合拆分申请回报	N/A	CUSTPFtdcTraderSpi::OnRtnMarginCombAction	私有流
	组合规则回报	N/A	CUSTPFtdcTraderSpi::OnRtnMarginCombinationLeg	私有流
询价回报	询价回报	N/A	CUSTPFtdcTraderSpi::OnRtnForQuote	询价流
查询	报单查询	CUSTPFtdcTraderApi::ReqQryOrder	CUSTPFtdcTraderSpi::OnRspQryOrder	查询流
	成交查询	CUSTPFtdcTraderApi::ReqQryTrade	CUSTPFtdcTraderSpi::OnRspQryTrade	查询流
	合约查询	CUSTPFtdcTraderApi::ReqQryInstrument	CUSTPFtdcTraderSpi::OnRspQryInstrument	查询流
	可用投资者查询	CUSTPFtdcTraderApi::ReqQryUserInvestor	CUSTPFtdcTraderSpi::OnRspQryUserInvestor	查询流

资金账户查询	CUSTPFtdcTraderApi::ReqQryInvestorAccount	CUSTPFtdcTraderSpi::OnRspQryInvestorAccount	查询流
交易编码查询	CUSTPFtdcTraderApi::ReqQryTradingCode	CUSTPFtdcTraderSpi::OnRspQryTradingCode	查询流
交易所查询	CUSTPFtdcTraderApi::ReqQryExchange	CUSTPFtdcTraderSpi::OnRspQryExchange	查询流
投资者持仓查询	CUSTPFtdcTraderApi::ReqQryInvestorPosition	CUSTPFtdcTraderSpi::OnRspQryInvestorPosition	查询流
投资者组合持仓查询	CUSTPFtdcTraderApi::ReqQryInvestorCombPosition	CUSTPFtdcTraderSpi::OnRspQryInvestorCombPosition	查询流
投资者单腿持仓查询	CUSTPFtdcTraderApi::ReqQryInvestorLegPosition	CUSTPFtdcTraderSpi::OnRspQryInvestorLegPosition	查询流
合规参数查询	CUSTPFtdcTraderApi::ReqQryComplianceParam	CUSTPFtdcTraderSpi::OnRspQryComplianceParam	查询流
交叉汇率查询	CUSTPFtdcTraderApi::ReqQryExchangeRate	CUSTPFtdcTraderSpi::OnRspQryExchangeRate	查询流
手续费率查询	CUSTPFtdcTraderApi::ReqQryInvestorFee	CUSTPFtdcTraderSpi::OnRspQryInvestorFee	查询流
保证金率查询	CUSTPFtdcTraderApi::ReqQryInvestorMargin	CUSTPFtdcTraderSpi::OnRspQryInvestorMargin	查询流

交易接口和私有流接口会有相互关联，如用户报单录入 ReqOrderInsert，马上会收到报单响应 OnRspOrderInsert，说明交易系统已经收到报单。报单进入交易系统后，如果报单的交易状态发生变化，就会收到报单回报 OnRtnOrder。如果报单被撮合(部分)成交，就会收到成交回报 OnRtnTrade。其中，一个用户的报单回报和成交回报也会被所属投资者下其他的用户接收到。

## 6. 开发接口

### 6.1 通用规则

客户端和飞马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 CUstpFtdcTraderApi 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, 订阅私有流
- 5, 订阅公共流
- 6, 设置飞马平台服务的地址

在功能调用阶段,程序可以任意调用交易接口中的请求方法,如 ReqOrderInsert 等。  
同时按照需要响应回调接口中的应答方法。

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。

### 6.2 CUstpFtdcTraderSpi 接口

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

#### 6.2.1 OnFrontConnected 方法

当客户端与飞马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。

#### 6.2.2 OnFrontDisconnected 方法

当客户端与飞马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。自动重连地址，可能是原来注册的地址，也可能是系统支持的其它可用的通信地址，它由程序自动选择。

**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

### 6.2.3 OnHeartBeatWarning 方法

心跳超时警告。当长时间未收到报文时，该方法被调用。

**函数原形：**

```
void OnHeartBeatWarning(int nTimeLapse);
```

**参数：**

nTimeLapse: 距离上次接收报文的时间

### 6.2.4 OnRspUserLogin 方法

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

**函数原形：**

```
void OnRspUserLogin(  
    CUstpFtdcRspUserLoginField *pRspUserLogin,  
    CUstpFtdcRspInfoField *pRspInfo,  
    int nRequestID,  
    bool bIsLast);
```

**参数：**

pRspUserLogin: 返回用户登录信息的地址。

用户登录信息结构：

```
struct CUstpFtdcRspUserLoginField  
{  
    ///交易日  
    TUstpFtdcDateType    TradingDay;  
    ///经纪公司编号  
    TUstpFtdcBrokerIDType    BrokerID;  
    ///交易用户代码
```



```

    TUstpFtdcUserIDType UserID;
    ///登录成功时间
    TUstpFtdcTimeType LoginTime;
    ///最大本地报单号
    TUstpFtdcOrderLocalIDType MaxOrderLocalID;
    ///交易系统名称
    TUstpFtdcTradingSystemNameType TradingSystemName;
};

```

pRspInfo: 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。

响应信息结构:

```

struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType 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。

### 6.2.5 OnRspUserLogout 方法

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

#### 函数原形:

```

void OnRspUserLogout(
    CUstpFtdcRspUserLogoutField *pRspUserLogout,
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast);

```

#### 参数:

pRspUserLogout: 返回用户退出信息的地址。

用户登出信息结构:

```

struct CUstpFtdcRspUserLogoutField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///交易用户代码
    TUstpFtdcUserIDType UserID;
};

```

pRspInfo: 返回用户响应信息的地址。

响应信息结构:

```

struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType 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。

## 6.2.6 OnRspUserPasswordUpdate 方法

用户密码修改应答。当客户端发出用户密码修改指令后, 飞马平台返回响应时, 该方法会被调用。

### 函数原形:

```

void OnRspUserPasswordUpdate(
    CUstpFtdcUserPasswordUpdateField *pUserPasswordUpdate,
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast);

```

### 参数:

pUserPasswordUpdate: 指向用户密码修改结构的地址, 包含了用户密码修改请求的输入数据。

用户密码修改结构:

```

struct CUstpFtdcUserPasswordUpdateField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///交易用户代码
    TUstpFtdcUserIDType UserID;
    ///旧密码
    TUstpFtdcPasswordType OldPassword;
    ///新密码
    TUstpFtdcPasswordType NewPassword;
};

```

pRspInfo: 返回用户响应信息的地址。

响应信息结构:

```

struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType ErrorMsg;
};

```

nRequestID: 返回用户密码修改请求的 ID, 该 ID 由用户在密码修改时指定。

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

### 6.2.7 OnRspError 方法

针对用户请求的出错通知。

#### 函数原形:

```
void OnRspError(  
    CUstpFtdcRspInfoField * pRspInfo,  
    int nRequestID,  
    bool bIsLast)
```

#### 参数:

pRspInfo: 返回用户响应信息的地址。

响应信息结构:

```
struct CUstpFtdcRspInfoField  
{  
    ///错误代码  
    TUstpFtdcErrorIDType ErrorID;  
    ///错误信息  
    TUstpFtdcErrorMsgType ErrorMsg;  
};
```

nRequestID: 返回用户操作请求的 ID, 该 ID 由用户在操作请求时指定。

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

### 6.2.8 OnRspOrderInsert 方法

报单录入应答。当客户端发出过报单录入指令后, 飞马平台返回响应时, 该方法会被调用。

#### 函数原形:

```
void OnRspOrderInsert(  
    CUstpFtdcInputOrderField *pInputOrder,  
    CUstpFtdcRspInfoField *pRspInfo,  
    int nRequestID,  
    bool bIsLast);
```

#### 参数:

pInputOrder: 指向报单录入结构的地址, 包含了提交报单录入时的输入数据, 和后台返回的报单编号。

输入报单结构:

```
struct CUstpFtdcInputOrderField  
{  
    ///经纪公司编号  
    TUstpFtdcBrokerIDType BrokerID;
```

```
///交易所代码
TUstpFtdcExchangeIDType ExchangeID;
///系统报单编号
TUstpFtdcOrderSysIDType OrderSysID;
///投资者编号
TUstpFtdcInvestorIDType InvestorID;
///用户代码
TUstpFtdcUserIDType UserID;
///指定下单席位编号（取值范围[1-N]，N为可用席位数目，超出范围的随机分配席位）
TUstpFtdcSeatNoType SeatNo;
///合约代码
TUstpFtdcInstrumentIDType InstrumentID;
///用户本地报单号
TUstpFtdcUserOrderLocalIDType UserOrderLocalID;
///报单类型
TUstpFtdcOrderPriceTypeType OrderPriceType;
///买卖方向
TUstpFtdcDirectionType Direction;
///开平标志
TUstpFtdcOffsetFlagType OffsetFlag;
///投机套保标志
TUstpFtdcHedgeFlagType HedgeFlag;
///价格
TUstpFtdcPriceType LimitPrice;
///数量
TUstpFtdcVolumeType Volume;
///有效期类型
TUstpFtdcTimeConditionType TimeCondition;
///GTD日期（暂不支持，保留域）
TUstpFtdcDateType GTDDate;
///成交量类型
TUstpFtdcVolumeConditionType VolumeCondition;
///最小成交量
TUstpFtdcVolumeType MinVolume;
///止损价（暂不支持，保留域）
TUstpFtdcPriceType StopPrice;
///强平原因（暂不支持，保留域）
TUstpFtdcForceCloseReasonType ForceCloseReason;
///自动挂起标志（暂不支持，保留域）
TUstpFtdcBoolType IsAutoSuspend;
///业务单元（暂不支持，保留域）
TUstpFtdcBusinessUnitType BusinessUnit;
///用户自定义域 64 字节
TUstpFtdcCustomType UserCustom;
```

```
};
```

pRspInfo: 返回用户响应信息的地址。

响应信息结构:

```
struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType ErrorMsg;
};
```

nRequestID: 返回报单录入操作请求的 ID, 该 ID 由用户在报单录入时指定。

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

### 6.2.9 OnRspOrderAction 方法

报单操作应答。报单操作包括报单的撤销、报单的挂起（暂不支持）、报单的激活（暂不支持）、报单的修改（暂不支持）。当客户端发出过报单操作指令后，飞马平台返回响应时，该方法会被调用。

**函数原形:**

```
void OnRspOrderAction(
    CUstpFtdcOrderActionField * pOrderAction,
    CUstpFtdcRspInfoField * pRspInfo,
    int nRequestID,
    bool bIsLast);
```

**参数:**

pOrderAction: 指向报单操作结构的地址, 包含了提交报单操作的输入数据, 和后台返回的报单编号。

报单操作结构:

```
struct CUstpFtdcOrderActionField
{
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///交易所系统报单编号
    TUstpFtdcOrderSysIDType OrderSysID;
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
    ///用户操作本地编号
    TUstpFtdcOrderLocalIDType UserOrderActionLocalID;
    ///本地报单编号
    TUstpFtdcUserOrderLocalIDType UserOrderLocalID;
```

```

        ///报单操作标志
        TUstpFtdcActionFlagType ActionFlag;
        ///价格（暂不支持，保留域）
        TUstpFtdcPriceType LimitPrice;
        ///数量变化（暂不支持，保留域）
        TUstpFtdcVolumeType VolumeChange;
};

```

pRspInfo: 返回用户响应信息的地址。

响应信息结构:

```

struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType ErrorMsg;
};

```

nRequestID: 返回用户报单操作请求的 ID, 该 ID 由用户在报单操作时指定。

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

#### 6.2.10 OnRspQryOrder 方法

报单查询请求。当客户端发出报单查询指令后, 飞马平台返回响应时, 该方法会被调用。

##### 函数原形:

```

void OnRspQryOrder(
    CUstpFtdcOrderField *pOrder,
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast);

```

##### 参数:

pOrder: 指向报单查询的返回结构信息的地址。

报单信息结构:

```

struct CUstpFtdcOrderField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///系统报单编号
    TUstpFtdcOrderSysIDType OrderSysID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
    ///用户代码

```

```
TUstpFtdcUserIDType UserID;
///指定下单席位序号（取值范围[1-N]，N为可用席位数目，超出范围的随机分配席位）
TUstpFtdcSeatNoType SeatNo;
///合约代码
TUstpFtdcInstrumentIDType InstrumentID;
///用户本地报单号
TUstpFtdcUserOrderLocalIDType UserOrderLocalID;
///报单类型
TUstpFtdcOrderPriceTypeType OrderPriceType;
///买卖方向
TUstpFtdcDirectionType Direction;
///开平标志
TUstpFtdcOffsetFlagType OffsetFlag;
///投机套保标志
TUstpFtdcHedgeFlagType HedgeFlag;
///价格
TUstpFtdcPriceType LimitPrice;
///数量
TUstpFtdcVolumeType Volume;
///有效期类型
TUstpFtdcTimeConditionType TimeCondition;
///GTD日期（暂不支持，保留域）
TUstpFtdcDateType GTDDate;
///成交量类型
TUstpFtdcVolumeConditionType VolumeCondition;
///最小成交量
TUstpFtdcVolumeType MinVolume;
///止损价（暂不支持，保留域）
TUstpFtdcPriceType StopPrice;
///强平原因（暂不支持，保留域）
TUstpFtdcForceCloseReasonType ForceCloseReason;
///自动挂起标志（暂不支持，保留域）
TUstpFtdcBoolType IsAutoSuspend;
///业务单元（暂不支持，保留域）
TUstpFtdcBusinessUnitType BusinessUnit;
///用户自定义域
TUstpFtdcCustomType UserCustom;
///交易日
TUstpFtdcTradingDayType TradingDay;
///会员编号
TUstpFtdcParticipantIDType ParticipantID;
///下单用户编号
TUstpFtdcUserIDType OrderUserID;
///客户号
```

```

    TUstpFtdcClientIDType   ClientID;
    ///下单席位号
    TUstpFtdcSeatIDType   SeatID;
    ///插入时间
    TUstpFtdcTimeType   InsertTime;
    ///本地报单编号
    TUstpFtdcOrderLocalIDType   OrderLocalID;
    ///报单来源
    TUstpFtdcOrderSourceType   OrderSource;
    ///报单状态
    TUstpFtdcOrderStatusType   OrderStatus;
    ///撤销时间
    TUstpFtdcTimeType   CancelTime;
    ///撤单用户编号
    TUstpFtdcUserIDType   CancelUserID;
    ///今成交数量
    TUstpFtdcVolumeType   VolumeTraded;
    ///剩余数量
    TUstpFtdcVolumeType   VolumeRemain;
};

```

pRspInfo: 返回用户响应信息的地址。

响应信息结构:

```

struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType   ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType   ErrorMsg;
};

```

nRequestID: 返回用户报单查询请求的 ID, 该 ID 由用户在报单查询时指定。

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

### 6.2.11 OnRspQryTrade 方法

成交单查询应答。当客户端发出成交单查询指令后, 飞马平台返回响应时, 该方法会被调用。

#### 函数原形:

```

void OnRspQryTrade(
    CUstpFtdcTradeField * pTrade,
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast);

```

#### 参数:



pTrade: 指向成交信息结构的地址。

成交信息结构:

```
struct CUstpFtdcTradeField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///交易日
    TUstpFtdcTradingDayType TradingDay;
    ///会员编号
    TUstpFtdcParticipantIDType ParticipantID;
    ///下单席位号
    TUstpFtdcSeatIDType SeatID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
    ///客户号
    TUstpFtdcClientIDType ClientID;
    ///用户编号
    TUstpFtdcUserIDType UserID;
    ///下单用户编号
    TUstpFtdcUserIDType OrderUserID;
    ///成交编号
    TUstpFtdcTradeIDType TradeID;
    ///本地报单编号
    TUstpFtdcOrderLocalIDType UserOrderLocalID;
    ///合约代码
    TUstpFtdcInstrumentIDType InstrumentID;
    ///买卖方向
    TUstpFtdcDirectionType Direction;
    ///开平标志
    TUstpFtdcOffsetFlagType OffsetFlag;
    ///投机套保标志
    TUstpFtdcHedgeFlagType HedgeFlag;
    ///成交价格
    TUstpFtdcTradePriceType TradePrice;
    ///成交数量
    TUstpFtdcTradeVolumeType TradeVolume;
    ///成交时间
    TUstpFtdcTradeTimeType TradeTime;
    ///清算会员编号
    TUstpFtdcParticipantIDType ClearingPartID;
    ///本次成交手续费
    TUstpFtdcMoneyType UsedFee;
```

```

    ///本次成交占用保证金
    TUstpFtdcMoneyType UsedMargin;
    ///本次成交占用权利金
    TUstpFtdcMoneyType Premium;
    ///持仓表今持仓量
    TUstpFtdcVolumeType Position;
    ///持仓表今日持仓成本
    TUstpFtdcPriceType PositionCost;
    ///资金表可用资金
    TUstpFtdcMoneyType Available;
    ///资金表占用保证金
    TUstpFtdcMoneyType Margin;
    ///资金表冻结的保证金
    TUstpFtdcMoneyType FrozenMargin;
};

```

pRspInfo: 指向响应信息结构的地址。

响应信息结构:

```

struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType ErrorMsg;
};

```

nRequestID: 返回用户成交单请求的 ID, 该 ID 由用户在成交单查询时指定。

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

### 6.2.12 OnRspQryInvestorAccount 方法

投资者资金账户查询。当客户端发出投资者资金账户查询指令后, 飞马平台返回响应时, 该方法会被调用。

#### 函数原形:

```

void OnRspQryInvestorAccount(
    CUstpFtdcRspInvestorAccountField *pRspInvestorAccount,
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast)

```

#### 参数:

pRspInvestorAccount 指向投资者资金账户信息的地址。

投资者账户结构

```
struct CUstpFtdcRspInvestorAccountField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType   BrokerID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
    ///资金帐号
    TUstpFtdcAccountIDType  AccountID;
    ///上次结算准备金
    TUstpFtdcMoneyType      PreBalance;
    ///入金金额
    TUstpFtdcMoneyType      Deposit;
    ///出金金额
    TUstpFtdcMoneyType      Withdraw;
    ///冻结的保证金
    TUstpFtdcMoneyType      FrozenMargin;
    ///冻结手续费
    TUstpFtdcMoneyType      FrozenFee;
    ///冻结权利金
    TUstpFtdcMoneyType      FrozenPremium;
    ///手续费
    TUstpFtdcMoneyType      Fee;
    ///平仓盈亏
    TUstpFtdcMoneyType      CloseProfit;
    ///持仓盈亏
    TUstpFtdcMoneyType      PositionProfit;
    ///可用资金
    TUstpFtdcMoneyType      Available;
    ///多头冻结的保证金
    TUstpFtdcMoneyType      LongFrozenMargin;
    ///空头冻结的保证金
    TUstpFtdcMoneyType      ShortFrozenMargin;
    ///多头占用保证金
    TUstpFtdcMoneyType      LongMargin;
    ///空头占用保证金
    TUstpFtdcMoneyType      ShortMargin;
    ///当日释放保证金
    TUstpFtdcMoneyType      ReleaseMargin;
    ///动态权益
    TUstpFtdcMoneyType      DynamicRights;
    ///今日出入金
    TUstpFtdcMoneyType      TodayInOut;
    ///占用保证金
    TUstpFtdcMoneyType      Margin;
```

```

    ///期权权利金收支
    TUstpFtdcMoneyType  Premium;
    ///风险度
    TUstpFtdcMoneyType  risk;
};

```

pRspInfo: 指向响应信息结构的地址。

响应信息结构:

```

struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType  ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType  ErrorMsg;
};

```

nRequestID: 返回用户成交单请求的 ID, 该 ID 由用户在成交单查询时指定。

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

### 6.2.13 OnRspQryTradingCode 方法

交易编码查询。当客户端发出交易编码查询指令后, 飞马平台返回响应时, 该方法会被调用。

#### 函数原形:

```

void OnRspQryTradingCode(
    CUstpFtdcRspTradingCodeField *pTradingCode,
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast)

```

#### 参数:

pTradingCode 指向交易编码查询返回信息的地址。

交易编码结构:

```

struct CUstpFtdcRspTradingCodeField
{
    ///交易所代码
    TUstpFtdcExchangeIDType  ExchangeID;
    ///经纪公司编号
    TUstpFtdcBrokerIDType  BrokerID;
    ///投资者编号
    TUstpFtdcInvestorIDType  InvestorID;
    ///客户代码
    TUstpFtdcClientIDType  ClientID;
    ///客户代码权限
    TUstpFtdcTradingRightType  ClientRight;
    ///是否活跃

```

```

    TUstpFtdcIsActiveType    IsActive;
};

```

pRspInfo: 指向响应信息结构的地址。

响应信息结构:

```

struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType ErrorMsg;
};

```

nRequestID: 返回用户成交单请求的 ID, 该 ID 由用户在成交单查询时指定。

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

#### 6.2.14 OnRspQryExchange 方法

交易所查询。当客户端发出交易所查询指令后, 飞马平台返回响应时, 该方法会被调用。

##### 函数原形:

```

void OnRspQryExchange(
    CUstpFtdcRspExchangeField *pExchange,
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast)

```

##### 参数:

pExchange 指向交易所结构的地址。

交易编码结构

```

struct CUstpFtdcRspExchangeField
{
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///交易所名称
    TUstpFtdcExchangeNameType ExchangeName;
    ///交易所状态
    TUstpFtdcExchangeStatusType ExchangeStatus;
};

```

pRspInfo: 指向响应信息结构的地址。

响应信息结构:

```

struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息

```

```
TUstpFtdcErrorMsgType ErrorMsg;
};
```

nRequestID: 返回用户成交单请求的 ID, 该 ID 由用户在成交单查询时指定。

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

### 6.2.15 OnRspQryInstrument 方法

合约信息查询应答。当客户端发出合约信息查询指令后, 飞马平台返回响应时, 该方法会被调用。

#### 函数原形:

```
void OnRspQryUstpInstrument(
    CUstpFtdcRspInstrumentField *pRspInstrument,
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast)
```

#### 参数:

pRspInstrument 指向合约信息结构的地址。

合约信息结构:

```
struct CUstpFtdcRspInstrumentField
{
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///品种代码
    TUstpFtdcProductIDType ProductID;
    ///品种名称
    TUstpFtdcProductNameType ProductName;
    ///合约代码
    TUstpFtdcInstrumentIDType InstrumentID;
    ///合约名称
    TUstpFtdcInstrumentNameType InstrumentName;
    ///交割年份
    TUstpFtdcDeliveryYearType DeliveryYear;
    ///交割月
    TUstpFtdcDeliveryMonthType DeliveryMonth;
    ///限价单最大下单量
    TUstpFtdcMaxLimitOrderVolumeType MaxLimitOrderVolume;
    ///限价单最小下单量
    TUstpFtdcMinLimitOrderVolumeType MinLimitOrderVolume;
    ///市价单最大下单量
    TUstpFtdcMaxMarketOrderVolumeType MaxMarketOrderVolume;
    ///市价单最小下单量
    TUstpFtdcMinMarketOrderVolumeType MinMarketOrderVolume;
    ///数量乘数
```

```

    TUstpFtdcVolumeMultipleType VolumeMultiple;
    ///报价单位
    TUstpFtdcPriceTickType PriceTick;
    ///币种
    TUstpFtdcCurrencyType Currency;
    ///多头限仓
    TUstpFtdcLongPosLimitType LongPosLimit;
    ///空头限仓
    TUstpFtdcShortPosLimitType ShortPosLimit;
    ///跌停板价
    TUstpFtdcLowerLimitPriceType LowerLimitPrice;
    ///涨停板价
    TUstpFtdcUpperLimitPriceType UpperLimitPrice;
    ///昨结算
    TUstpFtdcPreSettlementPriceType PreSettlementPrice;
    ///合约交易状态
    TUstpFtdcInstrumentStatusType InstrumentStatus;
    ///创建日
    TUstpFtdcDateType CreateDate;
    ///上市日
    TUstpFtdcDateType OpenDate;
    ///到期日
    TUstpFtdcDateType ExpireDate;
    ///开始交割日
    TUstpFtdcDateType StartDelivDate;
    ///最后交割日
    TUstpFtdcDateType EndDelivDate;
    ///挂牌基准价
    TUstpFtdcPriceType BasisPrice;
    ///当前是否交易
    TUstpFtdcBoolType IsTrading;
    ///基础商品代码
    TUstpFtdcInstrumentIDType UnderlyingInstrID;
    ///持仓类型
    TUstpFtdcPositionTypeType PositionType;
    ///执行价
    TUstpFtdcPriceType StrikePrice;
    ///期权类型
    TUstpFtdcOptionsTypeType OptionsType;
};
pRspInfo: 指向响应信息结构的地址。
响应信息结构:
struct CUstpFtdcRspInfoField
{

```

```

    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType ErrorMsg;
};

```

nRequestID: 返回用户成交单请求的 ID, 该 ID 由用户在成交单查询时指定。

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

## 6.2.16 OnRspQryUserInvestor 方法

可用投资者账户查询应答。当客户端发出可用投资者账户查询指令后, 飞马平台返回响应时, 该方法会被调用。

### 函数原形:

```

void OnRspQryUserInvestor(
    CUstpFtdcUserInvestorField *pUserInvestor,
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast)

```

### 参数:

pUserInvestor 指向用户可用投资者账户结构的地址。

投资者账户结构:

```

struct CUstpFtdcUserInvestorField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///交易用户代码
    TUstpFtdcUserIDType UserID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
};

```

pRspInfo: 指向响应信息结构的地址。

响应信息结构:

```

struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType ErrorMsg;
};

```

nRequestID: 返回用户成交单请求的 ID, 该 ID 由用户在成交单查询时指定。

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



### 6.2.17 OnRspQryInvestorPosition 方法

投资者持仓查询应答。当客户端发出投资者持仓查询指令后，飞马平台返回响应时，该方法会被调用。

#### 函数原形：

```
void OnRspQryInvestorPosition (
    CUstpFtdcRspInvestorPositionField *pInvestorPosition,
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast)
```

#### 参数：

pInvestorPosition 指向投资者持仓结构的地址。

投资者持仓结构：

```
struct CUstpFtdcRspInvestorPositionField
{
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///客户代码
    TUstpFtdcClientIDType ClientID;
    ///合约代码
    TUstpFtdcInstrumentIDType InstrumentID;
    ///买卖方向
    TUstpFtdcDirectionType Direction;
    ///投机套保标志
    TUstpFtdcHedgeFlagType HedgeFlag;
    ///占用保证金
    TUstpFtdcMoneyType UsedMargin;
    ///今持仓量
    TUstpFtdcVolumeType Position;
    ///今日持仓成本
    TUstpFtdcPriceType PositionCost;
    ///昨持仓量
    TUstpFtdcVolumeType YdPosition;
    ///昨日持仓成本
    TUstpFtdcMoneyType YdPositionCost;
    ///冻结的保证金
    TUstpFtdcMoneyType FrozenMargin;
    ///开仓冻结持仓
    TUstpFtdcVolumeType FrozenPosition;
```

```

    ///平仓冻结持仓
    TUstpFtdcVolumeType FrozenClosing;
    ///冻结的权利金
    TUstpFtdcMoneyType FrozenPremium;
    ///最后一笔成交编号
    TUstpFtdcTradeIDType LastTradeID;
    ///最后一笔本地报单编号
    TUstpFtdcOrderLocalIDType LastOrderLocalID;
    ///币种
    TUstpFtdcCurrencyIDType Currency;
};

```

pRspInfo: 指向响应信息结构的地址。

响应信息结构:

```

struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType ErrorMsg;
};

```

nRequestID: 返回用户成交单请求的 ID, 该 ID 由用户在成交单查询时指定。

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

## 6.2.18 OnRspQryComplianceParam 方法

合规参数查询应答。当客户端发出合规参数查询指令后, 飞马平台返回响应时, 该方法会被调用。

### 函数原形:

```

void OnRspQryComplianceParam (
    CUstpFtdcRspComplianceParamField *pRspComplianceParam,
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast)

```

### 参数:

pRspComplianceParam 指向合规参数结构的地址。

合规参数结构:

```

struct CUstpFtdcRspComplianceParamField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;

```

```

    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///客户号
    TUstpFtdcClientIDType ClientID;
    ///每日最大报单笔
    TUstpFtdcVolumeType DailyMaxOrder;
    ///每日最大撤单笔
    TUstpFtdcVolumeType DailyMaxOrderAction;
    ///每日最大错单笔
    TUstpFtdcVolumeType DailyMaxErrorOrder;
    ///每日最大报单手
    TUstpFtdcVolumeType DailyMaxOrderVolume;
    ///每日最大撤单手
    TUstpFtdcVolumeType DailyMaxOrderActionVolume;
};

```

pRspInfo: 指向响应信息结构的地址。

响应信息结构:

```

struct CUSTPFTDCRSPINFOFIELD
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType ErrorMsg;
};

```

nRequestID: 返回用户成交单请求的 ID, 该 ID 由用户在成交单查询时指定。

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

### 6.2.19 OnRspQryInvestorFee 方法

投资者手续费率查询应答。当客户端发出投资者手续费率查询指令后, 飞马平台返回响应时, 该方法会被调用。

#### 函数原形:

```

void OnRspQryInvestorFee(
    CUSTPFTDCINVESTORFEEFIELD *pInvestorFee,
    CUSTPFTDCRSPINFOFIELD *pRspInfo,
    int nRequestID,
    bool bIsLast)

```

#### 参数:

pInvestorFee 指向投资者手续费率结构的地址。

投资者手续费率结构:

```

struct CUstpFtdcInvestorFeeField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType   BrokerID;
    ///客户号
    TUstpFtdcClientIDType   ClientID;
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///合约代码
    TUstpFtdcInstrumentIDType InstrumentID;
    ///品种代码
    TUstpFtdcProductIDType  ProductID;
    ///开仓手续费按比例
    TUstpFtdcRatioType   OpenFeeRate;
    ///开仓手续费按手数
    TUstpFtdcRatioType   OpenFeeAmt;
    ///平仓手续费按比例
    TUstpFtdcRatioType   OffsetFeeRate;
    ///平仓手续费按手数
    TUstpFtdcRatioType   OffsetFeeAmt;
    ///平今仓手续费按比例
    TUstpFtdcRatioType   OTFeeRate;
    ///平今仓手续费按手数
    TUstpFtdcRatioType   OTFeeAmt;
    ///每笔订单收取的申报费
    TUstpFtdcRatioType   PerOrderAmt;
    ///每笔撤单收取的申报费
    TUstpFtdcRatioType   PerCancelAmt;
};

```

pRspInfo: 指向响应信息结构的地址。

响应信息结构:

```

struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType ErrorMsg;
};

```

nRequestID: 返回用户成交单请求的 ID, 该 ID 由用户在成交单查询时指定。

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

## 6.2.20 OnRspQryInvestorMargin 方法

投资者保证金率查询应答。当客户端发出投资者保证金率查询指令后, 飞马平台返

回响应时，该方法会被调用。

### 函数原形:

```
void OnRspQryInvestorMargin (
    CUstpFtdcInvestorMarginField *pInvestorMargin,
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast)
```

### 参数:

pInvestorMargin 指向投资者保证金率结构的地址。

投资者保证金率结构:

```
struct CUstpFtdcInvestorMarginField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType   BrokerID;
    ///客户号
    TUstpFtdcClientIDType   ClientID;
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///合约代码
    TUstpFtdcInstrumentIDType InstrumentID;
    ///品种代码
    TUstpFtdcProductIDType  ProductID;
    ///多头占用保证金按比例
    TUstpFtdcRatioType      LongMarginRate;
    ///多头保证金按手数
    TUstpFtdcRatioType      LongMarginAmt;
    ///空头占用保证金按比例
    TUstpFtdcRatioType      ShortMarginRate;
    ///空头保证金按手数
    TUstpFtdcRatioType      ShortMarginAmt;
};
```

pRspInfo: 指向响应信息结构的地址。

响应信息结构:

```
struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType ErrorMsg;
```

```
};
```

nRequestID: 返回用户成交单请求的 ID, 该 ID 由用户在成交单查询时指定。

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

### 6.2.21 OnRspQuoteInsert 方法

报价录入应答。当客户端发出报价录入指令后, 飞马平台返回响应时, 该方法会被调用。

#### 函数原形:

```
void OnRspQuoteInsert (
    CUstpFtdcInputQuoteField *pInputQuote,
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast)
```

#### 参数:

pInputQuote 指向报价结构的地址。

报价结构:

```
struct CUstpFtdcInputQuoteField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
    ///用户代码
    TUstpFtdcUserIDType UserID;
    ///指定通过此席位序号下单, 取值范围[1-N], N 为可用席位数目,
    //超出范围的随机分配席位
    TUstpFtdcSeatNoType SeatNo;
    ///合约代码
    TUstpFtdcInstrumentIDType InstrumentID;
    ///交易系统返回的系统报价编号
    TUstpFtdcQuoteSysIDType QuoteSysID;
    ///用户设定的本地报价编号
    TUstpFtdcUserQuoteLocalIDType UserQuoteLocalID;
    ///飞马向交易系统报的本地报价编号
    TUstpFtdcQuoteLocalIDType QuoteLocalID;
    ///买方买入数量
    TUstpFtdcVolumeTypeBidVolume;
    ///买方开平标志
    TUstpFtdcOffsetFlagType BidOffsetFlag;
```

```

    ///买方投机套保标志
    TUstpFtdcHedgeFlagType BidHedgeFlag;
    ///买方买入价格
    TUstpFtdcPriceType BidPrice;
    ///卖方卖出数量
    TUstpFtdcVolumeType AskVolume;
    ///卖方开平标志
    TUstpFtdcOffsetFlagType AskOffsetFlag;
    ///卖方投机套保标志
    TUstpFtdcHedgeFlagType AskHedgeFlag;
    ///卖方卖出价格
    TUstpFtdcPriceType AskPrice;
    ///业务单元
    TUstpFtdcBusinessUnitType BusinessUnit;
    ///用户自定义域
    TUstpFtdcCustomType UserCustom;
    ///拆分出来的买方用户本地报单编号
    TUstpFtdcUserOrderLocalIDType BidUserOrderLocalID;
    ///拆分出来的卖方用户本地报单编号
    TUstpFtdcUserOrderLocalIDType AskUserOrderLocalID;
};

```

报价录入时必须满足：MaxOrderLocalID < UserQuoteLocalID < BidUserOrderLocalID < AskUserOrderLocalID

pRspInfo: 指向响应信息结构的地址。

响应信息结构:

```

struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType ErrorMsg;
};

```

nRequestID: 返回用户报价请求的 ID, 该 ID 由用户在报价请求时指定。

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

## 6.2.22 OnRspQuoteAction 方法

报价操作应答。当客户端发出报价操作指令后, 飞马平台返回响应时, 该方法会被调用。

**函数原形:**

```

void OnRspQuoteAction (
    CustpFtdcQuoteActionField * pQuoteAction,
    CUstpFtdcRspInfoField * pRspInfo,

```

```
int nRequestID,
bool bIsLast)
```

**参数:**

pQuoteAction 指向报价操作结构的地址。

报价操作结构:

```
struct CUstpFtdcQuoteActionField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
    ///用户代码
    TUstpFtdcUserIDType UserID;
    ///交易系统返回的系统报价编号
    TUstpFtdcQuoteSysIDType QuoteSysID;
    ///用户设定的被撤的本地报价编号
    TUstpFtdcUserQuoteLocalIDType UserQuoteLocalID;
    ///用户向飞马报的本地撤消报价编号
    TUstpFtdcUserQuoteLocalIDType UserQuoteActionLocalID;
    ///报单操作标志
    TUstpFtdcActionFlagType ActionFlag;
    ///业务单元
    TUstpFtdcBusinessUnitType BusinessUnit;
    ///用户自定义域
    TUstpFtdcCustomType UserCustom;
};
```

pRspInfo: 指向响应信息结构的地址。

响应信息结构:

```
struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType ErrorMsg;
};
```

nRequestID: 返回用户报价操作请求的 ID, 该 ID 由用户在报价操作请求时指定。

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

**6.2.23 OnRspForQuote 方法**

询价应答。当客户端发出询价指令后, 飞马平台返回响应时, 该方法会被调用。



**函数原形:**

```
void OnRspForQuote(
    CUstpFtdcReqForQuoteField *pReqForQuote
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast)
```

**参数:**

pReqForQuote 指向询价结构的地址。

询价结构:

```
struct CUstpFtdcReqForQuoteField
{
    ///询价编号
    TUstpFtdcQuoteSysIDType ReqForQuoteID;
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
    ///用户代码
    TUstpFtdcUserIDType UserID;
    ///合约代码
    TUstpFtdcInstrumentIDType InstrumentID;
    ///交易日
    TUstpFtdcDateType TradingDay;
    ///询价时间
    TUstpFtdcTimeType ReqForQuoteTime;
};
```

pRspInfo: 指向响应信息结构的地址。

响应信息结构:

```
struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType ErrorMsg;
};
```

nRequestID: 返回询价请求的 ID, 该 ID 由用户在询价请求时指定。

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

**6.2.24 OnRspMarginCombAction 方法**

客户申请组合应答。当客户端发起组合操作请求 ReqMarginCombAction 时, 该方法

会被调用。

### 函数原形:

```
void OnRspMarginCombAction(
    CUstpFtdcInputMarginCombActionField *pInputMarginCombAction,
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast);
```

### 参数:

pInputMarginCombAction: 指向组合策略操作结构的地址

```
struct CUstpFtdcInputMarginCombActionField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType   BrokerID;
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///交易用户代码
    TUstpFtdcUserIDType   UserID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
    ///投机套保标志
    TUstpFtdcHedgeFlagType HedgeFlag;
    ///用户本地编号
    TUstpFtdcUserOrderLocalIDType   UserActionLocalID;
    ///组合合约代码
    TUstpFtdcInstrumentIDType   CombInstrumentID;
    ///组合数量
    TUstpFtdcVolumeType   CombVolume;
    ///组合动作方向
    TUstpFtdcCombDirectionType   CombDirection;
    ///本地编号
    TUstpFtdcOrderLocalIDType   ActionLocalID;
};
```

pRspInfo: 指向响应信息结构的地址。

响应信息结构:

```
struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType   ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType   ErrorMsg;
};
```

nRequestID: 返回询价请求的 ID, 该 ID 由用户在询价请求时指定。

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

### 6.2.25 OnRspUserDeposit 方法

投资者自助出入金应答。当客户端发起自助出入金请求 ReqUserDeposit 时，该方法会被调用。

#### 函数原形:

```
void OnRspUserDeposit(
    CUstpFtdcstpUserDepositField *pstpUserDeposit,
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast)
```

#### 参数:

pstpUserDeposit: 指向投资者自助出入金结构的地址

```
struct CUstpFtdcstpUserDepositField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///用户代码
    TUstpFtdcUserIDType UserID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
    ///金额
    TUstpFtdcMoneyType Amount;
    ///出入金方向
    TUstpFtdcAccountDirectionType AmountDirection;
    ///用户本地报单号
    TUstpFtdcUserOrderLocalIDType UserOrderLocalID;
};
```

pRspInfo: 指向响应信息结构的地址。

响应信息结构:

```
struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType ErrorMsg;
};
```

nRequestID: 返回询价请求的 ID, 该 ID 由用户在询价请求时指定。

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

### 6.2.1 OnRspTransferMoney 方法

投资者主次席出入金应答。当客户端发起主次席出入金请求 ReqTransferMoney 时，该方法会被调用。

#### 函数原形：

```
void OnRspTransferMoney (
    CUstpFtdcstpTransferMoneyField *pstpTransferMoney,
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast)
```

#### 参数：

pstpTransferMoney：指向投资者自助出入金结构的地址

```
struct CUstpFtdcstpTransferMoneyField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///用户代码
    TUstpFtdcUserIDType UserID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
    ///金额
    TUstpFtdcMoneyType Amount;
    ///出入金方向
    TUstpFtdcAccountDirectionType AmountDirection;
    ///用户本地报单号
    TUstpFtdcUserOrderLocalIDType UserOrderLocalID;
    ///银行机构代码
    TUstpFtdcBankIDType BankID;
    ///银行转账密码
    TUstpFtdcBankPasswdType BankPasswd;
    ///主席转账密码
    TUstpFtdcAccountPasswdType AccountPasswd;
};
```

pRspInfo：指向响应信息结构的地址。

#### 响应信息结构：

```
struct CUstpFtdcRspInfoField
```

```
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType ErrorMsg;
};
```

nRequestID: 返回请求的 ID, 该 ID 由用户在请求时指定。

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

### 6.2.2 OnRspQryInvestorLegPosition 方法

投资者单腿持仓查询应答。当客户端发出投资者单腿持仓查询指令后, 飞马平台返回响应时, 该方法会被调用。

#### 函数原形:

```
void OnRspQryInvestorLegPosition (
    CUstpFtdcRspInvestorLegPositionField *pInvestorPosition,
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast)
```

#### 参数:

pInvestorPosition 指向投资者单腿持仓结构的地址。

投资者单腿持仓结构:

```
struct CUstpFtdcRspInvestorLegPositionField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
    ///投机套保标志
    TUstpFtdcHedgeFlagType HedgeFlag;
    ///客户代码
    TUstpFtdcClientIDType ClientID;
    ///单腿合约代码
    TUstpFtdcInstrumentIDType InstrumentID;
    ///多头持仓
    TUstpFtdcVolumeType LongPosition;
```

```

    ///空头持仓
    TUstpFtdcVolumeType ShortPosition;
    ///多头占用保证金
    TUstpFtdcMoneyType LongMargin;
    ///空头占用保证金
    TUstpFtdcMoneyType ShortMargin;
    ///多头冻结持仓
    TUstpFtdcVolumeType LongFrozenPosition;
    ///空头冻结持仓
    TUstpFtdcVolumeType ShortFrozenPosition;
    ///多头冻结保证金
    TUstpFtdcMoneyType LongFrozenMargin;
    ///空头冻结保证金
    TUstpFtdcMoneyType ShortFrozenMargin;
};

```

pRspInfo: 指向响应信息结构的地址。

响应信息结构:

```

struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType ErrorMsg;
};

```

nRequestID: 返回用户成交单请求的 ID, 该 ID 由用户在成交单查询时指定。

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

### 6.2.3 OnRspQryInvestorCombPosition 方法

投资者组合持仓查询应答。当客户端发出投资者组合持仓查询指令后, 飞马平台返回响应时, 该方法会被调用。

#### 函数原形:

```

void OnRspQryInvestorCombPosition (
    CUstpFtdcRspInvestorCombPositionField *pInvestorPosition,
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast)

```

#### 参数:

pInvestorPosition 指向投资者组合持仓结构的地址。

投资者组合持仓结构:

```
struct CUstpFtdcRspInvestorCombPositionField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType   BrokerID;
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
    ///投机套保标志
    TUstpFtdcHedgeFlagType  HedgeFlag;
    ///客户代码
    TUstpFtdcClientIDType   ClientID;
    ///组合合约代码
    TUstpFtdcInstrumentIDType CombInstrumentID;
    ///组合持仓
    TUstpFtdcVolumeType      CombPosition;
    ///冻结组合持仓
    TUstpFtdcVolumeType      CombFrozenPosition;
    ///组合一手释放的保证金
    TUstpFtdcMoneyType       CombFreeMargin;
};
```

pRspInfo: 指向响应信息结构的地址。

响应信息结构:

```
struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType ErrorMsg;
};
```

nRequestID: 返回用户成交单请求的 ID, 该 ID 由用户在成交单查询时指定。

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

#### 6.2.4 OnRspQryExchangRate 方法

交叉汇率查询应答。当客户端发出交叉汇率查询指令后, 飞马平台返回响应时, 该方法会被调用。

**函数原形:**

```
void OnRspQryExchangeRate (
    CUstpFtdcRspExchangeRateField *pRspExchangeRate,
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast)
```

**参数:**

pExchange 指向交叉汇率结构的地址。

交叉汇率结构

```
struct CUstpFtdcRspExchangeRateField
{
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///品种代码
    TUstpFtdcProductIDType ProductID;
    ///折算汇率
    TUstpFtdcExchangeRateType ExchangeRate;
};
```

pRspInfo: 指向响应信息结构的地址。

响应信息结构:

```
struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType ErrorMsg;
};
```

nRequestID: 返回用户成交单请求的 ID, 该 ID 由用户在成交单查询时指定。

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

**6.2.5 OnRtnTrade 方法**

成交回报。当发生成交时飞马平台会通知客户端, 该方法会被调用。

**函数原形:**

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

**参数:**

pTrade: 指向成交信息结构的地址。

成交信息结构:

```
struct CUstpFtdcTradeField
{
```



```
///经纪公司编号
TUstpFtdcBrokerIDType BrokerID;
///交易所代码
TUstpFtdcExchangeIDType ExchangeID;
///交易日
TUstpFtdcTradingDayType TradingDay;
///会员编号
TUstpFtdcParticipantIDType ParticipantID;
///下单席位号
TUstpFtdcSeatIDType SeatID;
///投资者编号
TUstpFtdcInvestorIDType InvestorID;
///客户号
TUstpFtdcClientIDType ClientID;
///用户编号
TUstpFtdcUserIDType UserID;
///下单用户编号
TUstpFtdcUserIDType OrderUserID;
///成交编号
TUstpFtdcTradeIDType TradeID;
///本地报单编号
TUstpFtdcOrderLocalIDType UserOrderLocalID;
///合约代码
TUstpFtdcInstrumentIDType InstrumentID;
///买卖方向
TUstpFtdcDirectionType Direction;
///开平标志
TUstpFtdcOffsetFlagType OffsetFlag;
///投机套保标志
TUstpFtdcHedgeFlagType HedgeFlag;
///成交价格
TUstpFtdcTradePriceType TradePrice;
///成交数量
TUstpFtdcTradeVolumeType TradeVolume;
///成交时间
TUstpFtdcTradeTimeType TradeTime;
///清算会员编号
TUstpFtdcParticipantIDType ClearingPartID;
///本次成交手续费
TUstpFtdcMoneyType UsedFee;
///本次成交占用保证金
TUstpFtdcMoneyType UsedMargin;
///本次成交占用权利金
```

```

    TUstpFtdcMoneyType Premium;
    ///持仓表今持仓量
    TUstpFtdcVolumeType Position;
    ///持仓表今日持仓成本
    TUstpFtdcPriceType PositionCost;
    ///资金表可用资金
    TUstpFtdcMoneyType Available;
    ///资金表占用保证金
    TUstpFtdcMoneyType Margin;
    ///资金表冻结的保证金
    TUstpFtdcMoneyType FrozenMargin;
};

```

### 6.2.6 OnRtnOrder 方法

报单回报。当客户端进行报单录入、报单操作及其它原因（如部分成交）导致报单状态发生变化时，飞马平台会主动通知客户端，该方法会被调用。

#### 函数原形：

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

#### 参数：

pOrder：指向报单信息结构的地址。

报单信息结构：

```

struct CUstpFtdcOrderField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///系统报单编号
    TUstpFtdcOrderSysIDType OrderSysID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
    ///用户代码
    TUstpFtdcUserIDType UserID;
    ///指定下单席位编号（取值范围[1-N]，N为可用席位数目，超出范围的随机分配席位）
    TUstpFtdcSeatNoType SeatNo;
    ///合约代码
    TUstpFtdcInstrumentIDType InstrumentID;
    ///用户本地报单号

```

```
TUstpFtdcUserOrderLocalIDType  UserOrderLocalID;
///报单类型
TUstpFtdcOrderPriceTypeType  OrderPriceType;
///买卖方向
TUstpFtdcDirectionType  Direction;
///开平标志
TUstpFtdcOffsetFlagType  OffsetFlag;
///投机套保标志
TUstpFtdcHedgeFlagType  HedgeFlag;
///价格
TUstpFtdcPriceType  LimitPrice;
///数量
TUstpFtdcVolumeType  Volume;
///有效期类型
TUstpFtdcTimeConditionType  TimeCondition;
///GTD 日期（暂不支持，保留域）
TUstpFtdcDateType  GTDDate;
///成交量类型
TUstpFtdcVolumeConditionType  VolumeCondition;
///最小成交量
TUstpFtdcVolumeType  MinVolume;
///止损价（暂不支持，保留域）
TUstpFtdcPriceType  StopPrice;
///强平原因（暂不支持，保留域）
TUstpFtdcForceCloseReasonType  ForceCloseReason;
///自动挂起标志（暂不支持，保留域）
TUstpFtdcBoolType  IsAutoSuspend;
///业务单元（暂不支持，保留域）
TUstpFtdcBusinessUnitType  BusinessUnit;
///用户自定义域
TUstpFtdcCustomType  UserCustom;
///交易日
TUstpFtdcTradingDayType  TradingDay;
///会员编号
TUstpFtdcParticipantIDType  ParticipantID;
///最初下单用户编号
TUstpFtdcUserIDType  OrderUserID;
///客户号
TUstpFtdcClientIDType  ClientID;
///下单席位号
TUstpFtdcSeatIDType  SeatID;
///插入时间
TUstpFtdcTimeType  InsertTime;
///本地报单编号
```

```

    TUstpFtdcOrderLocalIDType    OrderLocalID;
    ///报单来源
    TUstpFtdcOrderSourceType      OrderSource;
    ///报单状态
    TUstpFtdcOrderStatusType      OrderStatus;
    ///撤销时间
    TUstpFtdcTimeType             CancelTime;
    ///撤单用户编号
    TUstpFtdcUserIDType           CancelUserID;
    ///今成交数量
    TUstpFtdcVolumeType           VolumeTraded;
    ///剩余数量
    TUstpFtdcVolumeType           VolumeRemain;
};

```

### 6.2.7 OnRtnInstrumentStatus 方法

合约交易状态通知。当合约状态发生变化时，由飞马平台主动通知客户端，该方法会被调用。

#### 函数原形：

```

void OnRtnInstrumentStatus(
    CUstpFtdcInstrumentStatusField *pInstrumentStatus) ;

```

#### 参数：

pInstrumentStatus：指向合约交易状态的地址，包含了后台返回的合约状态数据。

#### 参数结构：

```

struct CUstpFtdcInstrumentStatusField
{
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///品种代码
    TUstpFtdcProductIDType   ProductID;
    ///品种名称
    TUstpFtdcProductNameType  ProductName;
    ///合约代码
    TUstpFtdcInstrumentIDType InstrumentID;
    ///合约名称
    TUstpFtdcInstrumentNameType InstrumentName;
    ///交割年份
    TUstpFtdcDeliveryYearType DeliveryYear;
    ///交割月
    TUstpFtdcDeliveryMonthType DeliveryMonth;
    ///限价单最大下单量
    TUstpFtdcMaxLimitOrderVolumeType MaxLimitOrderVolume;
};

```

```
///限价单最小下单量
TUstpFtdcMinLimitOrderVolumeType    MinLimitOrderVolume;
///市价单最大下单量
TUstpFtdcMaxMarketOrderVolumeType    MaxMarketOrderVolume;
///市价单最小下单量
TUstpFtdcMinMarketOrderVolumeType    MinMarketOrderVolume;
///数量乘数
TUstpFtdcVolumeMultipleType    VolumeMultiple;
///报价单位
TUstpFtdcPriceTickType    PriceTick;
///币种
TUstpFtdcCurrencyType    Currency;
///多头限仓
TUstpFtdcLongPosLimitType    LongPosLimit;
///空头限仓
TUstpFtdcShortPosLimitType    ShortPosLimit;
///跌停板价
TUstpFtdcLowerLimitPriceType    LowerLimitPrice;
///涨停板价
TUstpFtdcUpperLimitPriceType    UpperLimitPrice;
///昨结算
TUstpFtdcPreSettlementPriceType    PreSettlementPrice;
///合约交易状态
TUstpFtdcInstrumentStatusType    InstrumentStatus;
///创建日
TUstpFtdcDateType    CreateDate;
///上市日
TUstpFtdcDateType    OpenDate;
///到期日
TUstpFtdcDateType    ExpireDate;
///开始交割日
TUstpFtdcDateType    StartDelivDate;
///最后交割日
TUstpFtdcDateType    EndDelivDate;
///挂牌基准价
TUstpFtdcPriceType    BasisPrice;
///当前是否交易
TUstpFtdcBoolType    IsTrading;
///基础商品代码
TUstpFtdcInstrumentIDType    UnderlyingInstrID;
///持仓类型
TUstpFtdcPositionTypeType    PositionType;
///执行价
TUstpFtdcPriceType    StrikePrice;
```

```

        ///期权类型
        TUstpFtdcOptionsTypeType    OptionsType;
};

```

### 6.2.8 OnRtnInvestorAccountDeposit 方法

出入金结果回报。当柜台端发起了出入金时，飞马平台主动通知客户端，该方法会被调用。

#### 函数原形：

```

void OnRtnInvestorAccountDeposit(
    CUstpFtdcInvestorAccountDepositResField* pInvestorAccountDepositRes)

```

#### 参数：

pInvestorAccountDepositRes：指向出入金结果回报结构的地址。

出入金结果回报结构：

```

struct CUstpFtdcInvestorAccountDepositResField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType    BrokerID;
    ///用户代码
    TUstpFtdcUserIDType    UserID;
    ///投资者编号
    TUstpFtdcInvestorIDType    InvestorID;
    ///资金帐号
    TUstpFtdcAccountIDType    AccountID;
    ///资金流水号
    TUstpFtdcAccountSeqNoType    AccountSeqNo;
    ///金额
    TUstpFtdcMoneyType    Amount;
    ///出入金方向
    TUstpFtdcAccountDirectionType    AmountDirection;
    ///可用资金
    TUstpFtdcMoneyType    Available;
    ///结算准备金
    TUstpFtdcMoneyType    Balance;
};

```

## 6.2.9 OnErrRtnOrderInsert 方法

报单录入错误回报。由飞马平台主动通知客户端，该方法会被调用。

### 函数原形：

```
void OnErrRtnOrderInsert(  
    CUstpFtdcInputOrderField *pInputOrder,  
    CUstpFtdcRspInfoField * pRspInfo);
```

### 参数：

pInputOrder：指向报单录入结构的地址，包含了提交报单录入时的输入数据，和后台返回的报单编号。

输入报单结构：

```
struct CUstpFtdcInputOrderField  
{  
    ///经纪公司编号  
    TUstpFtdcBrokerIDType BrokerID;  
    ///交易所代码  
    TUstpFtdcExchangeIDType ExchangeID;  
    ///系统报单编号  
    TUstpFtdcOrderSysIDType OrderSysID;  
    ///投资者编号  
    TUstpFtdcInvestorIDType InvestorID;  
    ///用户代码  
    TUstpFtdcUserIDType UserID;  
    ///指定下单席位编号（取值范围[1-N]，N为可用席位数目，超出范围的随机分配席位）  
    TUstpFtdcSeatNoType SeatNo;  
    ///合约代码  
    TUstpFtdcInstrumentIDType InstrumentID;  
    ///用户本地报单号  
    TUstpFtdcUserOrderLocalIDType UserOrderLocalID;  
    ///报单类型  
    TUstpFtdcOrderPriceTypeType OrderPriceType;  
    ///买卖方向  
    TUstpFtdcDirectionType Direction;  
    ///开平标志  
    TUstpFtdcOffsetFlagType OffsetFlag;  
    ///投机套保标志  
    TUstpFtdcHedgeFlagType HedgeFlag;  
    ///价格  
    TUstpFtdcPriceType LimitPrice;  
    ///数量  
    TUstpFtdcVolumeType Volume;  
    ///有效期类型
```

```

    TUstpFtdcTimeConditionType  TimeCondition;
    ///GTD 日期
    TUstpFtdcDateType    GTDDate;
    ///成交量类型
    TUstpFtdcVolumeConditionType    VolumeCondition;
    ///最小成交量
    TUstpFtdcVolumeType  MinVolume;
    ///止损价
    TUstpFtdcPriceType  StopPrice;
    ///强平原因
    TUstpFtdcForceCloseReasonType    ForceCloseReason;
    ///自动挂起标志
    TUstpFtdcBoolType    IsAutoSuspend;
    ///业务单元
    TUstpFtdcBusinessUnitType    BusinessUnit;
    ///用户自定义域 64 字节
    TUstpFtdcCustomType  UserCustom;
};

```

pRspInfo: 返回用户响应信息的地址。

响应信息结构:

```

struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType  ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType  ErrorMsg;
};

```

## 6.2.10 OnErrRtnOrderAction 方法

报单操作错误回报。由飞马平台主动通知客户端，该方法会被调用。

**函数原形:**

```

void OnErrRtnOrderAction (
    CUstpFtdcOrderActionField *pOrderAction,
    CUstpFtdcRspInfoField *pRspInfo);

```

**参数:**

pOrderAction: 指向报单操作结构的地址，包含了报单操作请求的输入数据。

报单操作结构:

```

struct CUstpFtdcOrderActionField
{
    ///交易所代码
    TUstpFtdcExchangeIDType  ExchangeID;
    ///交易所系统报单编号

```



```

    TUstpFtdcOrderSysIDType OrderSysID;
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
    ///用户操作本地编号
    TUstpFtdcOrderLocalIDType UserOrderActionLocalID;
    ///本地报单编号
    TUstpFtdcUserOrderLocalIDType UserOrderLocalID;
    ///报单操作标志
    TUstpFtdcActionFlagType ActionFlag;
    ///价格 - 保留域
    TUstpFtdcPriceType LimitPrice;
    ///数量变化-保留域
    TUstpFtdcVolumeType VolumeChange;
};
pRspInfo: 返回用户响应信息的地址。
响应信息结构:
struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType ErrorMsg;
};

```

### 6.2.11 OnRtnQuote 方法

报价回报。由飞马平台主动通知客户端，该方法会被调用。

#### 函数原形:

```
void OnRtnQuote (CUstpFtdcRtnQuoteField *pRtnQuote);
```

#### 参数:

pRtnQuote: 指向报价回报结构的地址，包含了报价操作请求的输入数据，和后台返回的报价编号。

报价回报结构:

```

struct CUstpFtdcRtnQuoteField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;

```

```
///用户代码
TUstpFtdcUserIDType UserID;
///合约代码
TUstpFtdcInstrumentIDType InstrumentID;
///交易系统返回的系统报价编号
TUstpFtdcQuoteSysIDType QuoteSysID;
///用户设定的本地报价编号
TUstpFtdcUserQuoteLocalIDType UserQuoteLocalID;
///飞马向交易系统报的本地报价编号
TUstpFtdcQuoteLocalIDType QuoteLocalID;
///买方买入数量
TUstpFtdcVolumeType BidVolume;
///买方开平标志
TUstpFtdcOffsetFlagType BidOffsetFlag;
///买方投机套保标志
TUstpFtdcHedgeFlagType BidHedgeFlag;
///买方买入价格
TUstpFtdcPriceType BidPrice;
///卖方卖出数量
TUstpFtdcVolumeType AskVolume;
///卖方开平标志
TUstpFtdcOffsetFlagType AskOffsetFlag;
///卖方投机套保标志
TUstpFtdcHedgeFlagType AskHedgeFlag;
///卖方卖出价格
TUstpFtdcPriceType AskPrice;
///业务单元
TUstpFtdcBusinessUnitType BusinessUnit;
///用户自定义域
TUstpFtdcCustomType UserCustom;
///拆分出来的买方用户本地报单编号
TUstpFtdcUserOrderLocalIDType BidUserOrderLocalID;
///拆分出来的卖方用户本地报单编号
TUstpFtdcUserOrderLocalIDType AskUserOrderLocalID;
///买方系统报单编号
TUstpFtdcQuoteSysIDType BidOrderSysID;
///卖方系统报单编号
TUstpFtdcQuoteSysIDType AskOrderSysID;
///报价单状态
TUstpFtdcQuoteStatusType QuoteStatus;
///插入时间
TUstpFtdcTimeType InsertTime;
///撤销时间
TUstpFtdcTimeType CancelTime;
```

```

    ///成交时间
    TUstpFtdcTimeType TradeTime;
};

```

### 6.2.12 OnErrRtnQuoteInsert 方法

报价录入错误回报。由飞马平台主动通知客户端，该方法会被调用。

#### 函数原形：

```

void OnErrRtnQuoteInsert (
    CUstpFtdcInputQuoteField *pInputQuote,
    CUstpFtdcRspInfoField *pRspInfo);

```

#### 参数：

pInputQuote: 指向报价录入错误回报结构的地址，包含了报价操作请求的输入数据。

报价录入错误回报结构：

```

struct CUstpFtdcInputQuoteField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
    ///用户代码
    TUstpFtdcUserIDType UserID;
    ///合约代码
    TUstpFtdcInstrumentIDType InstrumentID;
    ///交易系统返回的系统报价编号
    TUstpFtdcQuoteSysIDType QuoteSysID;
    ///用户设定的本地报价编号
    TUstpFtdcUserQuoteLocalIDType UserQuoteLocalID;
    ///飞马向交易系统报的本地报价编号
    TUstpFtdcQuoteLocalIDType QuoteLocalID;
    ///买方买入数量
    TUstpFtdcVolumeType BidVolume;
    ///买方开平标志
    TUstpFtdcOffsetFlagType BidOffsetFlag;
    ///买方投机套保标志
    TUstpFtdcHedgeFlagType BidHedgeFlag;
    ///买方买入价格
    TUstpFtdcPriceType BidPrice;
    ///卖方卖出数量
    TUstpFtdcVolumeType AskVolume;
    ///卖方开平标志

```

```

TUstpFtdcOffsetFlagType AskOffsetFlag;
///卖方投机套保标志
TUstpFtdcHedgeFlagType AskHedgeFlag;
///卖方卖出价格
TUstpFtdcPriceType AskPrice;
///业务单元
TUstpFtdcBusinessUnitType BusinessUnit;
///用户自定义域
TUstpFtdcCustomType UserCustom;
///拆分出来的买方用户本地报单编号
TUstpFtdcUserOrderLocalIDType BidUserOrderLocalID;
///拆分出来的卖方用户本地报单编号
TUstpFtdcUserOrderLocalIDType AskUserOrderLocalID;
};
pRspInfo: 返回用户响应信息的地址。
响应信息结构:
struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType ErrorMsg;
};

```

### 6.2.13 OnErrRtnQuoteAction 方法

报价操作错误回报。由飞马平台主动通知客户端，该方法会被调用。

#### 函数原形:

```

///报价撤单错误回报
void OnErrRtnQuoteAction(
    CUstpFtdcOrderActionField *pOrderAction,
    CUstpFtdcRspInfoField *pRspInfo)

```

#### 参数:

pOrderAction: 指向报价操作结构的地址，包含了报价操作请求的输入数据。

报价操作结构:

//报价操作

```

struct CUstpFtdcQuoteActionField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;

```

```

    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
    ///用户代码
    TUstpFtdcUserIDType UserID;
    ///交易系统返回的系统报价编号
    TUstpFtdcQuoteSysIDType QuoteSysID;
    ///用户设定的被撤的本地报价编号
    TUstpFtdcUserQuoteLocalIDType UserQuoteLocalID;
    ///用户向飞马报的本地撤消报价编号
    TUstpFtdcUserQuoteLocalIDType UserQuoteActionLocalID;
    ///报单操作标志
    TUstpFtdcActionFlagType ActionFlag;
    ///业务单元
    TUstpFtdcBusinessUnitType BusinessUnit;
    ///用户自定义域
    TUstpFtdcCustomType UserCustom;
};

```

pRspInfo: 返回用户响应信息的地址。

响应信息结构:

```

struct CUstpFtdcRspInfoField
{
    ///错误代码
    TUstpFtdcErrorIDType ErrorID;
    ///错误信息
    TUstpFtdcErrorMsgType ErrorMsg;
};

```

#### 6.2.14 OnRtnMarginCombinationLeg 方法

组合规则变化通知。当组合策略规则发生变化时，由飞马平台主动通知客户端，该方法会被调用。

**函数原形:**

```

void OnRtnMarginCombinationLeg(
    CUstpFtdcMarginCombinationLegField *pMarginCombinationLeg);

```

**参数:**

pMarginCombinationLeg: 指向组合策略规则结构的地址。

组合策略规则结构:

```

    ///组合规则
    struct CUstpFtdcMarginCombinationLegField
    {
        ///交易所代码
        TUstpFtdcExchangeIDType ExchangeID;
    };

```

```

///组合合约代码
TUstpFtdcInstrumentIDType  CombInstrumentID;
///单腿编号
TUstpFtdcLegIDType  LegID;
///单腿合约代码
TUstpFtdcInstrumentIDType  LegInstrumentID;
///买卖方向
TUstpFtdcDirectionType  Direction;
///单腿乘数
TUstpFtdcLegMultipleType  LegMultiple;
///优先级
TUstpFtdcPriorityType  Priority;
};

```

### 6.2.15 OnRtnMarginCombAction 方法

#### 函数原形:

```

void OnRtnMarginCombAction(
    CUstpFtdcInputMarginCombActionField * pInputMarginCombAction);

```

#### 参数:

pInputMarginCombAction: 指向组合操作请求结构的地址。

组合操作请求结构:

```

struct CUstpFtdcInputMarginCombActionField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType  BrokerID;
    ///交易所代码
    TUstpFtdcExchangeIDType  ExchangeID;
    ///交易用户代码
    TUstpFtdcUserIDType  UserID;
    ///投资者编号
    TUstpFtdcInvestorIDType  InvestorID;
    ///投机套保标志
    TUstpFtdcHedgeFlagType  HedgeFlag;
    ///用户本地编号
    TUstpFtdcUserOrderLocalIDType  UserActionLocalID;
    ///组合合约代码
    TUstpFtdcInstrumentIDType  CombInstrumentID;
    ///组合数量
    TUstpFtdcVolumeType  CombVolume;
    ///组合动作方向
    TUstpFtdcCombDirectionType  CombDirection;
    ///本地编号

```

```
TUstpFtdcOrderLocalIDType   ActionLocalID;
};
```

### 6.2.16 OnRtnUserDeposit 方法

客户自助出入金通知接口，当客户自动出入金成功后由飞马平台主动推送

**函数原形：**

```
void OnRtnUserDeposit(CUstpFtdcstpUserDepositField *pstpUserDeposit);
```

**参数：**

pstpUserDeposit：指向自助出入金请求结构的地址。

自助出入金请求结构：

```
struct CUstpFtdcstpUserDepositField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType   BrokerID;
    ///用户代码
    TUstpFtdcUserIDType   UserID;
    ///投资者编号
    TUstpFtdcInvestorIDType   InvestorID;
    ///金额
    TUstpFtdcMoneyType   Amount;
    ///出入金方向
    TUstpFtdcAccountDirectionType   AmountDirection;
    ///用户本地报单号
    TUstpFtdcUserOrderLocalIDType   UserOrderLocalID;
};
```

## 6.3 CUSTPFtdcTraderApi 接口

CUstpFtdcTraderApi 接口提供给用户的功能包括，报单的录入、报单的撤销、报单的查询、成交单查询、客户持仓查询、合约查询、合约交易状态查询、交易所公告查询等功能。

### 6.3.1 CreateFtdcTraderApi 方法

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

**函数原形：**

```
static CUstpFtdcTradeApi *CreateFtdcTradeApi(const char *pszFlowPath = "");
```

**参数：**

pszFlowPath：常量字符指针，用于指定一个文件目录来存贮飞马平台发布消息的状态。默认值

代表当前目录。

返回值:

返回一个指向 CUSTpFtdcTradeApi 实例的指针。

### 6.3.2 Release 方法

释放一个 CUSTpFtdcTradeApi 实例。不能使用 delete 方法

**函数原形:**

```
void Release();
```

### 6.3.3 Init 方法

使客户端开始与飞马平台建立连接，连接成功后可以进行登陆。

**函数原形:**

```
void Init();
```

### 6.3.4 Join 方法

客户端等待一个接口实例线程的结束。

**函数原形:**

```
void Join();
```

### 6.3.5 GetTradingDay 方法

获得当前交易日。只有当与飞马平台连接建立后才会取到正确的值。

**函数原形:**

```
const char *GetTradingDay();
```

返回值:

返回一个指向日期信息字符串的常量指针。

### 6.3.6 RegisterSpi 方法

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

**函数原形:**

```
void RegisterSpi(CUSTpFtdcTraderSpi *pSpi);
```

**参数:**

pSpi: 实现了 CUSTpFtdcTraderSpi 接口的实例指针。



### 6.3.7 RegisterFront 方法

设置飞马平台的网络通讯地址，飞马平台拥有多个通信地址，但用户只需要选择一个通信地址。

#### 函数原形：

```
void RegisterFront(char *pszFrontAddress);
```

#### 参数：

pszFrontAddress：指向后台服务器地址的指针。

服务器地址的格式为：“protocol://ipaddress:port”，如：“tcp://127.0.0.1:17001”。“tcp”代表传输协议，“127.0.0.1”代表服务器地址。“17001”代表服务器端口号。

### 6.3.8 RegisterNameServer 方法

设置飞马 NameServer 的网络通讯地址，用于获取行情服务列表。交易系统拥有多个 NameServer，用户可以同时注册多个 NameServer 的网络通讯地址。

该方法要在 Init 方法之前调用。

#### 函数原型：

```
void RegisterNameServer (char *pszNsAddress);
```

#### 参数：

**pszNsAddress**：指向飞马服务端 NameServer 网络通讯地址的指针。网络地址的格式为：“protocol://ipaddress:port”，如：“tcp://127.0.0.1:17001”。“tcp”代表传输协议，“127.0.0.1”代表服务器地址。“17001”代表服务器端口号。

注意：此接口保留，但目前并未启用！

### 6.3.9 SubscribePrivateTopic 方法

订阅私有流。该方法要在 Init 方法前调用。若不调用则不会收到私有流的数据。

#### 函数原形：

```
void SubscribePrivateTopic(USTP_TE_RESUME_TYPE nResumeType);
```

#### 参数：

nResumeType：私有流重传方式

USTP\_TERT\_RESTART：从本交易日开始重传

USTP\_TERT\_RESUME：从上次收到的续传

USTP\_TERT\_QUICK：只传送登录后私有流的内容

### 6.3.10 SubscribePublicTopic 方法

订阅公共流。该方法要在 Init 方法前调用。若不调用则不会收到公共流的数据。

**函数原形：**

```
void SubscribePublicTopic(USTP_TE_RESUME_TYPE nResumeType);
```

**参数：**

nResumeType: 公共流重传方式

USTP\_TERT\_RESTART: 从本交易日开始重传

USTP\_TERT\_RESUME: 从上次收到的续传

USTP\_TERT\_QUICK: 只传送登录后公共流的内容

### 6.3.11 SubscribeForQuote 方法

订阅询价流。该方法要在 Init 方法前调用。若不调用则不会收到询价流的数据。

**函数原形：**

```
void SubscribeForQuote(USTP_TE_RESUME_TYPE nResumeType);
```

**参数：**

nResumeType: 公共流重传方式

USTP\_TERT\_RESTART: 从本交易日开始重传

USTP\_TERT\_RESUME: 从上次收到的续传

USTP\_TERT\_QUICK: 只传送登录后公共流的内容

### 6.3.12 ReqUserLogin 方法

用户发出登陆请求。

**函数原形：**

```
int ReqUserLogin (  
    CUstpFtdcReqUserLoginField *pReqUserLogin,  
    int nRequestID);
```

**参数：**

pReqUserLogin: 指向用户登录请求结构的地址。

用户登录请求结构：

```
struct CUstpFtdcReqUserLoginField  
{  
    ///交易日，API 自动填写  
    TUstpFtdcDateType    TradingDay;  
    ///交易用户代码
```

```

    TUstpFtdcUserIDType UserID;
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///密码
    TUstpFtdcPasswordType Password;
    ///用户端产品信息
    TUstpFtdcProductInfoType UserProductInfo;
    ///接口端产品信息一占位字段, API 自动填写
    TUstpFtdcProductInfoType InterfaceProductInfo;
    ///IP 地址一占位字段, API 自动填写
    TUstpFtdcIPAddressType IPAddress;
    ///Mac 地址一占位字段, API 自动填写
    TUstpFtdcMacAddressType MacAddress;
    ///协议信息一占位字段, API 自动填写
    TUstpFtdcProtocolInfoType ProtocolInfo;
};

```

nRequestID: 用户登录请求的 ID, 该 ID 由用户指定, 管理。

用户需要填写 UserProductInfo 字段, 即客户端的产品信息, 如软件开发商、版本号等。  
InterfaceProductInfo 和 ProtocolInfo 只须占位, 不必有效赋值。

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。

### 6.3.13 ReqUserLogout 方法

用户发出登出请求。

**函数原形:**

```

int ReqUserLogout (
    CUstpFtdcReqUserLogoutField * pReqUserLogout,
    int nRequestID);

```

**参数:**

pReqUserLogout: 指向用户登出请求结构的地址。

用户登出请求结构:

```

struct CUstpFtdcReqUserLogoutField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///交易用户代码
    TUstpFtdcUserIDType UserID;
};

```

### 6.3.14 ReqUserPasswordUpdate 方法

用户密码修改请求。

#### 函数原形：

```
int ReqUserPasswordUpdate (
    CUstpFtdcUserPasswordUpdateField *pUserPasswordUpdate,
    int nRequestID);
```

#### 参数：

pUserPasswordUpdate：指向用户口令修改结构的地址。

用户口令修改结构：

```
struct CUstpFtdcUserPasswordUpdateField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType   BrokerID;
    ///交易用户代码
    TUstpFtdcUserIDType   UserID;
    ///旧密码
    TUstpFtdcPasswordType   OldPassword;
    ///新密码
    TUstpFtdcPasswordType   NewPassword;
};
```

### 6.3.15 ReqOrderInsert 方法

客户端发出报单录入请求。

#### 函数原形：

```
int ReqOrderInsert (
    CUstpFtdcInputOrderField *pInputOrder,
    int nRequestID);
```

#### 参数：

pInputOrder：指向输入报单结构的地址。

输入报单结构：

```
struct CUstpFtdcInputOrderField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType   BrokerID;
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///系统报单编号
    TUstpFtdcOrderSysIDType OrderSysID;
    ///投资者编号（客户资金账户）
    TUstpFtdcInvestorIDType InvestorID;
```

```

///用户代码（飞马用户代码）
TUstpFtdcUserIDType UserID;
///指定下单席位编号（取值范围[1-N]，N为可用席位数目，超出范围的随机分配席位）
TUstpFtdcSeatNoType SeatNo;
///合约代码
TUstpFtdcInstrumentIDType InstrumentID;
///用户本地报单号
TUstpFtdcUserOrderLocalIDType UserOrderLocalID;
///报单价格条件（支持市价、限价、五档价和最优价）
TUstpFtdcOrderPriceTypeType OrderPriceType;
///买卖方向
TUstpFtdcDirectionType Direction;
///开平标志（只支持开仓和平仓，不支持强平、平昨、平今）
TUstpFtdcOffsetFlagType OffsetFlag;
///投机套保标志（支持投机、套保、套利、做市商编码）
TUstpFtdcHedgeFlagType HedgeFlag;
///价格
TUstpFtdcLimitPriceType LimitPrice;
///数量
TUstpFtdcVolumeType Volume;
///有效期类型（“立即完成，否则撤销”1 OR “当日有效”3）
TUstpFtdcTimeConditionType TimeCondition;
///GTD日期（暂不支持，保留域）
TUstpFtdcDateType GTDDate;
///成交量类型（支持“任意数量”和“全部数量”）
TUstpFtdcVolumeConditionType VolumeCondition;
///最小成交量（暂不支持，保留域）
TUstpFtdcVolumeType MinVolume;
///止损价（暂不支持，保留域）
TUstpFtdcPriceType StopPrice;
///强平原因，只支持“非强平”
TUstpFtdcForceCloseReasonType ForceCloseReason;
///自动挂起标志（暂不支持，保留域）
TUstpFtdcBoolType IsAutoSuspend;
///业务单元（暂不支持，保留域）
TUstpFtdcBusinessUnitType BusinessUnit;
///用户自定义域 64 字节
TUstpFtdcCustomType UserCustom;
};

```

**注意:**

*用户本地报单号 UserOrderLocalID 是一个 21 位的字符串，下一笔报单的本地报单编号需要比前一笔报单的本地报单编号大（不一定需要连续），其比较方式为字符串的比较。*

*$MaxOrderLocalID < UserOrderLocalID$*

**普通限价单必须填充的字段包括：**

- 1) BrokerID, 会员号, 形如 “2008”;
- 2) ExchangeID, 交易所代码, 只支持 “CFFEX”
- 3) InvestorID, 投资者编号, 形如 “10000029”;
- 4) UserID, 用户代码, 形如 “test1”;
- 5) InstrumentID, 合约代码, 形如 “IF1109”;
- 6) **OrderPriceType, 报单价格条件, 只能为 USTP\_FTDC\_OPT\_LimitPrice;**
- 7) Direction, 买卖方向, USTP\_FTDC\_D\_Buy 表示买, USTP\_FTDC\_D\_Sell 表示卖;
- 8) OffsetFlag, 开平标志, “0” 表示开仓, “1” 表示平仓;
- 9) HedgeFlag, 投机套保标志, “1” 为投机, “2” 为套保, “3” 为套利, “4” 为做市商;
- 10) LimitPrice, 价格, 形如 3500.00;
- 11) Volume, 数量, 例如 5 表示 5 手;
- 12) **TimeCondition, 有效期类型, 为 USTP\_FTDC\_TC\_GFD (“当日有效”);**
- 13) VolumeCondition, 成交量类型, 为 USTP\_FTDC\_VC\_AV (“任意数量”);
- 14) ContingentCondition, 触发条件, 为 USTP\_FTDC\_CC\_Immediately (“立即”);
- 15) ForceCloseReason, 强平原因, 为 USTP\_FTDC\_FCC\_NotForceClose (“非强平”);
- 16) UserOrderLocalID, 用户本地报单编号, 形如 “00000025”。

**普通市价单必须填充的字段包括：**

- 1) BrokerID, 会员号, 形如 “2008”;
- 2) ExchangeID, 交易所代码, 只支持 “CFFEX”
- 3) InvestorID, 客户号, 形如 “10000029”;
- 4) UserID, 交易用户代码, 形如 “test1”;
- 5) InstrumentID, 合约代码, 形如 “IF1109”;
- 6) **OrderPriceType, 报单价格条件, 只能为 USTP\_FTDC\_OPT\_AnyPrice;**
- 7) Direction, 买卖方向, USTP\_FTDC\_D\_Buy 表示买, USTP\_FTDC\_D\_Sell 表示卖;
- 8) OffsetFlag, 开平标志, “0” 表示开仓, “1” 表示平仓;
- 9) HedgeFlag, 投机套保标志, “1” 为投机, “2” 为套保, “3” 为套利, “4” 为做市商;
- 10) Volume, 数量, 例如 5 表示 5 手;
- 11) **TimeCondition, 有效期类型, 只能为 USTP\_FTDC\_TC\_IOC (“立即成交, 否则撤销”);**
- 12) VolumeCondition, 成交量类型, 只能为 USTP\_FTDC\_VC\_AV (“任意数量”);

13) ContingentCondition, 触发条件, 只能为 USTP\_FTDC\_CC\_Immediately (“立即”);

14) ForceCloseReason, 强平原因, 只能为 USTP\_FTDC\_FCC\_NotForceClose (“非强平”);

15) UserOrderLocalID, 本地报单编号, 形如“00000025”。

**其他 7 种订单类型, 详细说明如下:**

订单类型	OrderPriceType	TimeCondition	VolumeCondition	备注
FOK	USTP_FTDC_OPT_LimitPrice	USTP_FTDC_TC_IOC	USTP_FTDC_VC_CV	立即全部成交否则自动撤销
FAK	USTP_FTDC_OPT_LimitPrice	USTP_FTDC_TC_IOC	USTP_FTDC_VC_AV	立即成交剩余自动撤销
市价转限价	USTP_FTDC_OPT_AnyPrice	USTP_FTDC_TC_GFD	USTP_FTDC_VC_AV	市价单未成交部分转为最新价限价单
五档市价	USTP_FTDC_OPT_FiveLevelPrice	USTP_FTDC_TC_IOC	USTP_FTDC_VC_AV	市价单与对手方五档价格报单尝试成交, 剩余未成交部分撤销
五档市价转限价	USTP_FTDC_OPT_FiveLevelPrice	USTP_FTDC_TC_GFD	USTP_FTDC_VC_AV	市价单与对手方五档价格报单尝试成交, 剩余未成交部分转为最新价限价单
最优价	USTP_FTDC_OPT_BestPrice	USTP_FTDC_TC_IOC	USTP_FTDC_VC_AV	市价单与对手方最优一档价格报单尝试成交, 剩余未成交部分撤销
最优价转限价	USTP_FTDC_OPT_BestPrice	USTP_FTDC_TC_GFD	USTP_FTDC_VC_AV	市价单与对手方最优一档价格报单尝试成交, 剩余未成交部分转为最新价限价单

*FOK 单和 FAK 单为限价单, 除这 3 个字段外的其他字段填充方式参照限价单。*

*其余 5 种订单为市价单, 除这 3 个字段外的其他字段填充方式参照市价单。*

### 6.3.16 ReqOrderAction 方法

客户端发出报单操作请求, 包括报单的撤销、报单的挂起 (暂不支持)、报单的激活 (暂不支持)、报单的修改 (暂不支持)

**函数原形:**

```
int ReqOrderAction(
    CUSTPFTDCOrderActionField *pOrderAction,
    int nRequestID);
```

**参数:**

pOrderAction: 指向报单操作结构的地址。

报单操作结构:

```
struct CUSTPFTDCOrderActionField
{
    ///交易所代码
    TUSTPFTDCExchangeIDType ExchangeID;
    ///交易所系统报单编号 (不为空按该字段操作, 为空按本地报单编号操作)
```

```

TUstpFtdcOrderSysIDType OrderSysID;
///经纪公司编号
TUstpFtdcBrokerIDType BrokerID;
///投资者编号
TUstpFtdcInvestorIDType InvestorID;
///用户代码
TUstpFtdcUserIDType UserID;
///指定下单席位编号（取值范围[1-N]，N为可用席位数目，超出范围的随机分配席位）
TUstpFtdcSeatNoType SeatNo;
///本次撤单 Req 的本地编号，需要保证按字典序递增
TUstpFtdcUserOrderLocalIDType UserOrderActionLocalID;
///被撤订单的本地报单编号
TUstpFtdcUserOrderLocalIDType UserOrderLocalID;
///报单操作标志
TUstpFtdcActionFlagType ActionFlag;
///价格（暂不支持，保留域）
TUstpFtdcPriceType LimitPrice;
///数量变化（暂不支持，保留域）
TUstpFtdcVolumeType VolumeChange;
};

```

nRequestID: 用户报单操作请求的 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。

#### 撤单必须填充的字段包括:

- 1) BrokerID, 会员号, 形如“2008”;
- 2) ExchangeID, 交易所代码, 形如“CFFEX”;
- 3) InvestorID, 投资者编号, 形如“10000029”;
- 4) UserID, 用户代码, 形如“test1”;
- 5) OrderSysID 或者 UserOrderLocalID 任选其一, 被撤订单的订单编号, 优先按照 OrderSysID 定位, 形如“00000025”;
- 6) UserOrderActionLocalID, 撤单请求的本地编号, 与 UserOrderLocalID 同一个序列按字典序增加, 形如“00000025”;
- 7) ActionFlag, 操作标志, 只能填 **USTP\_FTDC\_AF\_Delete** (“删除”);



### 6.3.17 ReqUserDeposit 方法

客户端发出自助出入金请求，以调整飞马系统的资金额度。

**注意：**只有期货公司指定客户才有权限使用本接口。

**函数原形：**

///用户请求出入金

```
int ReqUserDeposit(  
    CUstpFtdcstpUserDepositField *pstpUserDeposit,  
    int nRequestID);
```

**参数：**

pstpUserDeposit：指向自助出入金结构的地址。

自助出入金结构：

```
struct CUstpFtdcstpUserDepositField  
{  
    ///经纪公司编号  
    TUstpFtdcBrokerIDType BrokerID;  
    ///用户代码  
    TUstpFtdcUserIDType UserID;  
    ///投资者编号  
    TUstpFtdcInvestorIDType InvestorID;  
    ///金额  
    TUstpFtdcMoneyType Amount;  
    ///出入金方向  
    TUstpFtdcAccountDirectionType AmountDirection;  
    ///用户本地报单号  
    TUstpFtdcUserOrderLocalIDType UserOrderLocalID;  
};
```

nRequestID：用户报单操作请求的 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。

**自助出入金必须填充的字段包括：**

- 1) BrokerID，会员号，形如“2008”；
- 2) InvestorID，投资者编号，形如“10000029”；
- 3) UserID，用户代码，形如“test1”；
- 4) UserOrderLocalID，与报单类、报价类、组合类接口的操作编号保持一个序列增加，形如“00000027”；
- 5) Amount，资金调整金额，形如“50000”；

- 6) AmountDirection, 出入金方向, **USTP\_FTDC\_AD\_In (入金)** 和 **USTP\_FTDC\_AD\_Out (出金)**;

### 6.3.18 ReqTransferMoney 方法

客户端发出主次席出入金请求, 由终端投资者发起, 飞马对接期货公司主席, 主席判断是否出入金成功, 飞马系统根据主席的判断来调整客户余额。

#### 函数原形:

///主次席出入金

```
int ReqTransferMoney (
    CUSTpFtdcstpTransferMoneyField *pstpTransferMoney,
    int nRequestID);
```

#### 参数:

pstpTransferMoney: 指向主次席出入金结构的地址。

主次席出入金结构:

```
struct CUSTpFtdcstpTransferMoneyField
{
    ///经纪公司编号
    TUSTpFtdcBrokerIDType BrokerID;
    ///用户代码
    TUSTpFtdcUserIDType UserID;
    ///投资者编号
    TUSTpFtdcInvestorIDType InvestorID;
    ///金额
    TUSTpFtdcMoneyType Amount;
    ///出入金方向
    TUSTpFtdcAccountDirectionType AmountDirection;
    ///用户本地报单号
    TUSTpFtdcUserOrderLocalIDType UserOrderLocalID;
    ///银行机构代码
    TUSTpFtdcBankIDType BankID;
    ///银行转账密码
    TUSTpFtdcBankPasswdType BankPasswd;
    ///主席转账密码
    TUSTpFtdcAccountPasswdType AccountPasswd; };
nRequestID: 用户报单操作请求的 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。

### 6.3.19 ReqQryOrder 方法

报单查询请求。

#### 函数原形:

```
int ReqQryUstpOrder (  
    CUstpFtdcQryOrderField *pQryOrder,  
    int nRequestID);
```

#### 参数:

pQryOrder: 指向报单查询结构的地址。

报单查询结构:

```
struct CUstpFtdcQryOrderField  
{  
    ///经纪公司编号  
    TUstpFtdcBrokerIDType   BrokerID;  
    ///用户代码  
    TUstpFtdcUserIDType   UserID;  
    ///交易所代码  
    TUstpFtdcExchangeIDType ExchangeID;  
    ///投资者编号  
    TUstpFtdcInvestorIDType InvestorID;  
    ///报单编号  
    TUstpFtdcOrderSysIDType OrderSysID;  
    ///合约代码  
    TUstpFtdcInstrumentIDType InstrumentID;  
    ///报单状态  
    TUstpFtdcOrderStatusType OrderStatus;  
};
```

### 6.3.20 ReqQryTrade 方法

成交单查询请求。

#### 函数原形:

```
int ReqQryUstpTrade (  
    CUstpFtdcQryTradeField *pQryTrade,  
    int nRequestID);
```

#### 参数:

pQryTrade: 指向成交查询结构的地址。

成交查询结构:

```
struct CUstpFtdcQryTradeField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType   BrokerID;
    ///用户代码
    TUstpFtdcUserIDType   UserID;
    ///交易所代码
    TUstpFtdcExchangeIDType   ExchangeID;
    ///投资者编号
    TUstpFtdcInvestorIDType   InvestorID;
    ///成交编号
    TUstpFtdcTradeIDType   TradeID;
    ///合约代码
    TUstpFtdcInstrumentIDType   InstrumentID;
};
```

### 6.3.21 ReqQryInvestorAccount 方法

投资者资金账户查询。

**函数原形:**

```
int ReqQryInvestorAccount(
    CUstpFtdcQryInvestorAccountField *pQryInvestorAccount,
    int nRequestID)
```

**参数:**

pQryInvestorAccount 指向投资者账户查询结构的地址。

投资者账户查询结构:

```
struct CUstpFtdcQryInvestorAccountField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType   BrokerID;
    ///用户代码
    TUstpFtdcUserIDType   UserID;
    ///投资者编号
    TUstpFtdcInvestorIDType   InvestorID;
};
```

### 6.3.22 ReqQryTradingCode 方法

交易编码查询。

**函数原形:**

```
int ReqQryTradingCode (
```

```
CUstpFtdcQryTradingCodeField *pQryTradingCode,  
int nRequestID)
```

**参数:**

pQryTradingCode 指向交易编码查询结构的地址。

交易编码结构:

```
struct CUstpFtdcQryTradingCodeField  
{  
    ///经纪公司编号  
    TUstpFtdcBrokerIDType BrokerID;  
    ///用户代码  
    TUstpFtdcUserIDType UserID;  
    ///投资者编号  
    TUstpFtdcInvestorIDType InvestorID;  
    ///交易所代码  
    TUstpFtdcExchangeIDType ExchangeID;  
};
```

### 6.3.23 ReqQryExchange 方法

交易所查询。

**函数原形:**

```
int ReqQryExchange(  
    CUstpFtdcQryExchangeField *pQryExchange,  
    int nRequestID)
```

**参数:**

pQryExchange 指向交易所查询结构的地址。

交易编码结构:

```
struct CUstpFtdcQryExchangeField  
{  
    ///交易所代码  
    TUstpFtdcExchangeIDType ExchangeID;  
};
```

### 6.3.24 ReqQryInstrument 方法

合约信息查询。

**函数原形:**

```
int ReqQryInstrument (  
    CUstpFtdcQryInstrumentField *pQryInstrument,  
    int nRequestID)
```

**参数:**

pQryInstrument 指向合约信息查询结构的地址。

合约查询结构

```
struct CUstpFtdcQryInstrumentField
{
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///产品代码
    TUstpFtdcProductIDType ProductID;
    ///合约代码
    TUstpFtdcInstrumentIDType InstrumentID;
};
```

### 6.3.25 ReqQryUserInvestor 方法

可用投资者账户查询。

**函数原形：**

```
int ReqQryUserInvestor(
    CUstpFtdcQryUserInvestorField *pQryUserInvestor,
    int nRequestID)
```

**参数：**

pQryUserInvestor 指向可用投资者账户查询结构的地址。

可用投资者查询结构

```
struct CUstpFtdcQryUserInvestorField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///用户代码
    TUstpFtdcUserIDType UserID;
};
```

### 6.3.26 ReqQryInvestorPosition 方法

投资者持仓查询。

**函数原形：**

```
int ReqQryInvestorPosition (
    CUstpFtdcQryInvestorPositionField *pQryUserInvestorPosition,
    int nRequestID)
```

**参数：**

pQryUserInvestorPosition 指向投资者持仓查询结构的地址。

投资者持仓查询结构

```
struct CUstpFtdcQryInvestorPositionField
{
    ///经纪公司编号
```

```

    TUstpFtdcBrokerIDType   BrokerID;
    ///用户代码
    TUstpFtdcUserIDType   UserID;
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
    ///合约代码
    TUstpFtdcInstrumentIDType InstrumentID;
};

```

### 6.3.27 ReqQryComplianceParam 方法

合规参数查询。

#### 函数原形:

```

int ReqQryComplianceParam (
    CUstpFtdcQryComplianceParamField *pQryComplianceParam,
    int nRequestID)

```

#### 参数:

pQryComplianceParam 指向合规参数查询结构的地址。

合规参数查询结构

```

struct CUstpFtdcQryComplianceParamField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType   BrokerID;
    ///用户代码
    TUstpFtdcUserIDType   UserID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
};

```

### 6.3.28 ReqQryInvestorFee 方法

投资者手续费率查询。

#### 函数原形:

```

int ReqQryInvestorFee (
    CUstpFtdcQryInvestorFeeField *pQryInvestorFee,
    int nRequestID)

```

#### 参数:

pQryInvestorFee 指向投资者手续费率查询结构的地址。

投资者手续费率查询结构

```
struct CUstpFtdcQryInvestorFeeField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType   BrokerID;
    ///用户代码
    TUstpFtdcUserIDType   UserID;
    ///投资者编号
    TUstpFtdcInvestorIDType   InvestorID;
    ///交易所代码
    TUstpFtdcExchangeIDType   ExchangeID;
    ///合约代码
    TUstpFtdcInstrumentIDType   InstrumentID;
};
```

### 6.3.29 ReqQryInvestorMargin 方法

投资者保证金率查询

**函数原形:**

```
int ReqQryInvestorMargin (
    CUstpFtdcQryInvestorMarginField *pQryInvestorMargin,
    int nRequestID)
```

**参数:**

pQryInvestorMargin 指向投资者保证金率查询结构的地址。

投资者保证金率查询结构

```
struct CUstpFtdcQryInvestorMarginField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType   BrokerID;
    ///用户代码
    TUstpFtdcUserIDType   UserID;
    ///投资者编号
    TUstpFtdcInvestorIDType   InvestorID;
    ///交易所代码
    TUstpFtdcExchangeIDType   ExchangeID;
    ///合约代码
    TUstpFtdcInstrumentIDType   InstrumentID;
};
```

### 6.3.30 ReqQuoteInsert 方法

报价录入请求



做市商进行双边报价的接口

### 函数原形:

```
int ReqQuoteInsert (  
    CUstpFtdcInputQuoteField *pInputQuote,  
    int nRequestID)
```

### 参数:

pInputQuote 指报价录入结构的地址。

报价录入结构

```
struct CUstpFtdcInputQuoteField  
{  
    ///经纪公司编号  
    TUstpFtdcBrokerIDType BrokerID;  
    ///交易所代码  
    TUstpFtdcExchangeIDType ExchangeID;  
    ///投资者编号  
    TUstpFtdcInvestorIDType InvestorID;  
    ///用户代码  
    TUstpFtdcUserIDType UserID;  
    ///合约代码  
    TUstpFtdcInstrumentIDType InstrumentID;  
    ///交易系统返回的系统报价编号  
    TUstpFtdcQuoteSysIDType QuoteSysID;  
    ///用户设定的本地报价编号  
    TUstpFtdcUserQuoteLocalIDType UserQuoteLocalID;  
    ///飞马向交易系统报的本地报价编号  
    TUstpFtdcQuoteLocalIDType QuoteLocalID;  
    ///买方买入数量  
    TUstpFtdcVolumeType BidVolume;  
    ///买方开平标志  
    TUstpFtdcOffsetFlagType BidOffsetFlag;  
    ///买方投机套保标志  
    TUstpFtdcHedgeFlagType BidHedgeFlag;  
    ///买方买入价格  
    TUstpFtdcPriceType BidPrice;  
    ///卖方卖出数量  
    TUstpFtdcVolumeType AskVolume;  
    ///卖方开平标志  
    TUstpFtdcOffsetFlagType AskOffsetFlag;  
    ///卖方投机套保标志  
    TUstpFtdcHedgeFlagType AskHedgeFlag;  
    ///卖方卖出价格  
    TUstpFtdcPriceType AskPrice;  
    ///业务单元  
    TUstpFtdcBusinessUnitType BusinessUnit;
```

```

    ///用户自定义域
    TustpFtdcCustomType UserCustom;
    ///拆分出来的买方用户本地报单编号
    TustpFtdcUserOrderLocalIDType BidUserOrderLocalID;
    ///拆分出来的卖方用户本地报单编号
    TustpFtdcUserOrderLocalIDType AskUserOrderLocalID;
    ///下单席位号
    TustpFtdcSeatIDType SeatID;
};

```

**报价单**必须填充的字段包括:

- 1) BrokerID, 会员号, 形如 “2008”;
- 2) ExchangeID, 交易所代码, 只支持 “CFFEX”
- 3) InvestorID, 投资者编号, 形如 “10000029”;
- 4) UserID, 用户代码, 形如 “test1”;
- 5) InstrumentID, 合约代码, 形如 “IF1109”;
- 6) **UserQuoteLocalID, 用户本地报价单号, 形如 “000000101”;**
- 7) BidVolume, 买方买入数量, 例如 5 表示 5 手;
- 8) BidOffsetFlag, 买方开平标志, “0” 表示开仓, “1” 表示平仓;
- 9) BidHedgeFlag, 买方投机套保标志, 必须填 “4”, 表示用做市商编码;
- 10) BidPrice, 价格, 形如 3510.00;
- 11) AskVolume, 买方买入数量, 例如 5 表示 5 手;
- 12) AskOffsetFlag, 买方开平标志, “0” 表示开仓, “1” 表示平仓;
- 13) AskHedgeFlag, 买方投机套保标志, 必须填 “4”, 表示用做市商编码;
- 14) AskPrice, 价格, 形如 3500.00;
- 15) **BidUserOrderLocalID, 报价单中买单的用户本地报单号, 例如 “000000102”;**
- 16) **AskUserOrderLocalID, 报价单中卖单的用户本地报单号, 例如 “000000103”。**

***注意:** MaxOrderLocalID < UserQuoteLocalID < BidUserOrderLocalID < AskUserOrderLocalID*

### 6.3.31 ReqQuoteAction 方法

报价操作请求

撤销某一笔报价, 仅做市商有权限进行此操作。

**函数原形:**

```
int ReqQuoteAction (
```

```
CUstpFtdcQuoteActionField *pQuoteAction,
int nRequestID)
```

### 参数:

pQuoteAction 指报价操作结构的地址。

报价操作结构

```
struct CUstpFtdcQuoteActionField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
    ///用户代码
    TUstpFtdcUserIDType UserID;
    ///交易系统返回的系统报价编号
    TUstpFtdcQuoteSysIDType QuoteSysID;
    ///用户设定的被撤的本地报价编号
    TUstpFtdcUserQuoteLocalIDType UserQuoteLocalID;
    ///用户向飞马报的本地撤消报价编号
    TUstpFtdcUserQuoteLocalIDType UserQuoteActionLocalID;
    ///报单操作标志
    TUstpFtdcActionFlagType ActionFlag;
    ///业务单元
    TUstpFtdcBusinessUnitType BusinessUnit;
    ///用户自定义域
    TUstpFtdcCustomType UserCustom;
};
```

**撤销报价单**必须填充的字段包括:

- 1) BrokerID, 会员号, 形如 “2008”;
- 2) ExchangeID, 交易所代码, 只支持 “CFFEX”
- 3) InvestorID, 投资者编号, 形如 “10000029”;
- 4) UserID, 用户代码, 形如 “test1”;
- 5) InstrumentID, 合约代码, 形如 “IF1109”;
- 6) QuoteSysID /UserQuoteLocalID, 要撤销报价单的系统报价单号/用户本地报价单号, 形如 “000000101”。二者选其一即可, 优先按照系统报价单号撤销;
- 7) UserQuoteActionLocalID, 撤销报价操作的用户本地操作编号, 形如 “000000101”;

**注意:** *MaxOrderLocalID < UserQuoteActionLocalID*

### 6.3.32 ReqForQuote 方法

#### 询价请求

由客户发起的询价请求。针对某一个合约询价后，做市商的报价在该合约的行情上反映。

#### 函数原形：

```
int ReqForQuote (  
    CUstpFtdcReqForQuoteField *pReqForQuote,  
    int nRequestID)
```

#### 参数：

pReqForQuote 指向询价结构的地址。

询价结构

```
struct CUstpFtdcReqForQuoteField  
{  
    ///询价编号  
    TUstpFtdcQuoteSysIDType ReqForQuoteID;  
    ///经纪公司编号  
    TUstpFtdcBrokerIDType BrokerID;  
    ///交易所代码  
    TUstpFtdcExchangeIDType ExchangeID;  
    ///投资者编号  
    TUstpFtdcInvestorIDType InvestorID;  
    ///用户代码  
    TUstpFtdcUserIDType UserID;  
    ///合约代码  
    TUstpFtdcInstrumentIDType InstrumentID;  
    ///交易日  
    TUstpFtdcDateType TradingDay;  
    ///询价时间  
    TUstpFtdcTimeType ReqForQuoteTime;  
};
```

询价单必须填充的字段包括：

- 1) BrokerID, 会员号, 形如“2008”;
- 2) ExchangeID, 交易所代码, 只支持“CFFEX”
- 3) InvestorID, 投资者编号, 形如“10000029”;
- 4) UserID, 用户代码, 形如“test1”;
- 5) InstrumentID, 合约代码, 形如“IF1109”;

### 6.3.33 ReqMarginCombAction 方法

策略组合/策略拆分请求

此接口用于对已有持仓进行组合，或者对已有组合进行拆分。该接口只能对中金所公布的策略组合规则表中的组合合约进行组合。

**函数原形：**

```
int ReqMarginCombAction(  
    CUstpFtdcInputMarginCombActionField *pInputMarginCombAction,  
    int nRequestID)
```

**参数：**

pInputMarginCombAction 指向策略组合结构的地址。

询价结构

```
struct CUstpFtdcInputMarginCombActionField  
{  
    ///经纪公司编号  
    TUstpFtdcBrokerIDType BrokerID;  
    ///交易所代码  
    TUstpFtdcExchangeIDType ExchangeID;  
    ///交易用户代码  
    TUstpFtdcUserIDType UserID;  
    ///投资者编号  
    TUstpFtdcInvestorIDType InvestorID;  
    ///投机套保标志  
    TUstpFtdcHedgeFlagType HedgeFlag;  
    ///用户本地编号  
    TUstpFtdcUserOrderLocalIDType UserActionLocalID;  
    ///组合合约代码  
    TUstpFtdcInstrumentIDType CombInstrumentID;  
    ///组合数量  
    TUstpFtdcVolumeType CombVolume;  
    ///组合动作方向  
    TUstpFtdcCombDirectionType CombDirection;  
    ///本地编号  
    TUstpFtdcOrderLocalIDType ActionLocalID;  
};
```

**策略组合/策略拆分必须填充的字段包括：**

- 1) BrokerID, 会员号, 形如 “2008”;
- 2) ExchangeID, 交易所代码, 只支持 “CFFEX”
- 3) InvestorID, 投资者编号, 形如 “10000029”;
- 4) UserID, 用户代码, 形如 “test1”;
- 5) HedgeFlag, 合约代码, 形如 “IF1109”;
- 6) UserActionLocalID, 用户本地报价单号, 形如 “000000101”;
- 7) CombInstrumentID, 组合合约代码, 形如 “2213”;
- 8) CombVolume, 组合数量, 形如 “1”;
- 9) CombDirection, 组合方向。“USTP\_FTDC\_CA\_Combine”为组合, “USTP\_FTDC\_CA\_Uncombine”为拆分组合

### 6.3.34 ReqQryInvestorCombPosition 方法

投资者组合持仓查询。

**函数原形:**

```
int ReqQryInvestorCombPosition(
    CUstpFtdcQryInvestorCombPositionField *pQryInvestorCombPosition,
    int nRequestID)
```

**参数:**

pQryUserInvestorPosition 指向投资者组合持仓查询结构的地址。

投资者组合持仓查询结构

```
struct CUstpFtdcQryInvestorPositionField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType BrokerID;
    ///用户代码
    TUstpFtdcUserIDType UserID;
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
    ///组合合约代码
    TUstpFtdcInstrumentIDType CombInstrumentID;
};
```

### 6.3.35 ReqQryInvestorLegPosition 方法

投资者单腿持仓查询。

**函数原形:**

```
int ReqQryInvestorLegPosition (
    CUstpFtdcQryInvestorLegPositionField *pQryInvestorLegPosition,
    int nRequestID)
```

**参数:**

pQryInvestorLegPosition 指向投资者单腿持仓查询结构的地址。

投资者单腿持仓查询结构

```
struct CUstpFtdcQryInvestorLegPositionField
{
    ///经纪公司编号
    TUstpFtdcBrokerIDType   BrokerID;
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///投资者编号
    TUstpFtdcInvestorIDType InvestorID;
    ///投机套保标志
    TUstpFtdcHedgeFlagType  HedgeFlag;
    ///单腿合约代码
    TUstpFtdcInstrumentIDType LegInstrumentID;
};
```

**6.3.36 ReqQryExchangeRate 方法**

交叉汇率查询。

注意: 外汇期货合约专用接口, 目前尚未启用。

**函数原形:**

```
int ReqQryExchangeRate(
    CUstpFtdcQryExchangeRateField *pQryExchangeRate,
    int nRequestID)
```

**参数:**

pQryExchangeRate 指向交叉汇率查询结构的地址。

交叉汇率查询结构

```
struct CUstpFtdcQryExchangeRateField
{
    ///交易所代码
    TUstpFtdcExchangeIDType ExchangeID;
    ///品种代码
    TUstpFtdcProductIDType  ProductID;
};
```

## 7. 开发实例

```
// testtradeapi.cpp :
// 一个简单的例子，介绍 CUSTPFtdcTraderApi 和 CUSTPFtdcTraderSpi 接口的使用。
// 本例将演示一个报单录入操作的过程
#include <stdio.h>
#include <windows.h>
#include "USTPFtdcTraderApi.h"
// 报单录入操作是否完成的标志
// Create a manual reset event with no signal
HANDLE g_hEvent = CreateEvent(NULL, true, false, NULL);
// 经纪公司代码
TUSTPFtdcBrokerIDType g_ch BrokerID;
// 交易用户代码
TUSTPFtdcUserIDType g_chUserID;
//用户本地最大报单号
int g_UserOrderLocalID;
class CSimpleHandler : public CUSTPFtdcTraderSpi
{
public:
// 构造函数，需要一个有效的指向 CUSTPFtdcMduserApi 实例的指针
CSimpleHandler(CUSTPFtdcTraderApi *pUserApi) :
m_pUserApi (pUserApi) {}
~CSimpleHandler () {}
// 当客户端与飞马平台建立起通信连接，客户端需要进行登录
virtual void OnFrontConnected()
{
    CUSTPFtdcReqUserLoginField reqUserLogin;
    // get BrokerID
    printf("BrokerID:");
    scanf("%s", &g_chBrokerID);
    strcpy(reqUserLogin. BrokerID, g_chBrokerID);
    // get userid
    printf("userid:");
    scanf("%s", &g_chUserID);
    strcpy(reqUserLogin.UserID, g_chUserID);
    // get password
    printf("password:");
    scanf("%s", &reqUserLogin.Password);
    // 发出登陆请求
    m_pUserApi->ReqUserLogin(&reqUserLogin, 0);
}
// 当客户端与飞马平台通信连接断开时，该方法被调用
```



```
virtual void OnFrontDisconnected(int nReason)
{
    // 当发生这个情况后, API 会自动重新连接, 客户端可不做处理
    printf("OnFrontDisconnected. \n");
}
// 当客户端发出登录请求之后, 该方法会被调用, 通知客户端登录是否成功
virtual void OnRspUserLogin(
    CUstpFtdcRspUserLoginField *pRspUserLogin,
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast)
{
    printf("OnRspUserLogin:\n");
    printf("ErrorCode=[%d], ErrorMessage=[%s]\n",
        pRspInfo->ErrorID, pRspInfo->ErrorMsg);
    printf("RequestID=[%d], Chain=[%d]\n", nRequestID, bIsLast);
    if (pRspInfo->ErrorID != 0)
    {
        // 端登失败, 客户端需进行错误处理
        printf("Failed to login, errorcode=%d errmsg=%s
            requestid=%d chain=%d",
            pRspInfo->ErrorID, pRspInfo->ErrorMsg,
            nRequestID, bIsLast);
        exit(-1);
    }
    //用户最大本地报单号
    g_UserOrderLocalID=atoi(pRspUserLogin->MaxOrderLocalID)+1;
    // 端登成功, 发出报单录入请求
    CUstpFtdcInputOrderField ord;
    memset(&ord, 0, sizeof(ord));
    //经纪公司代码
    strcpy(ord.BrokerID, g_chBrokerID);
    // 合约代码
    strcpy(ord.InstrumentID, "IF1306");
    //投资者代码
    strcpy(ord.InvestorID, "000101");
    // 用户代码
    strcpy(ord.UserID, g_chUserID);
    // 买卖方向
    strcpy(ord.Direction, "0");
    // 开平标志
    strcpy(ord.OffsetFlag, "0");
    // 投机套保标志
    strcpy(ord.HedgeFlag, "1");
```

```

// 价格
    ord.LimitPrice = 5000;
// 数量
    ord.VolumeTotalOriginal = 10;
// 有效期类型
    strcpy(ord.TimeCondition, "1");
// 自动挂起标志
    ord.IsAutoSuspend = 0;
//交易所
    strcpy(ord.ExchangeID, "CFFEX");
//本地报单号
    sprintf(ord.UserOrderLocalID, "%012d", g_UserOrderLocalID++);
    m_pUserApi->ReqOrderInsert(&ord, 1);
}
// 报单录入应答
virtual void OnRspOrderInsert(
    CUstpFtdcInputOrderField *pInputOrder,
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast)
{
// 输出报单录入结果
    printf("ErrorCode=[%d], ErrorMsg=[%s]\n",
        pRspInfo->ErrorID, pRspInfo->ErrorMsg);
// 通知报单录入完成
    SetEvent(g_hEvent);
};
///报单回报
virtual void OnRtnOrder(CUstpFtdcOrderField *pOrder)
{
    printf("OnRtnOrder:\n");
    printf("OrderSysID=[%s]\n", pOrder->OrderSysID);
}
// 针对用户请求的出错通知
virtual void OnRspError(
    CUstpFtdcRspInfoField *pRspInfo,
    int nRequestID,
    bool bIsLast)
{
    printf("OnRspError:\n");
    printf("ErrorCode=[%d], ErrorMsg=[%s]\n",
        pRspInfo->ErrorID, pRspInfo->ErrorMsg);
    printf("RequestID=[%d], Chain=[%d]\n", nRequestID, bIsLast);
// 客户端需进行错误处理

```

```
        {
            //客户端的错误处理
        }
    }
private:
    // 指向 CUSTpFtdcMduserApi 实例的指针
    CUSTpFtdcTraderApi *m_pUserApi;
};
int main()
{
    // 产生一个 CUSTpFtdcTraderApi 实例
    CUSTpFtdcTraderApi *pUserApi =
        CUSTpFtdcTraderApi::CreateFtdcTraderApi();
    // 产生一个事件处理的实例
    CSimpleHandler sh(pUserApi);
    // 注册一事件处理的实例
    pUserApi->RegisterSpi(&sh);
    // 订阅私有流
    // USTP_TERT_RESTART:从本交易日开始重传
    // USTP_TERT_RESUME:从上次收到的续传
    // USTP_TERT_QUICK:只传送登录后私有流的内容
    pUserApi->SubscribePrivateTopic(USTP_TERT_RESUME);
    // 订阅公共流
    // USTP_TERT_RESTART:从本交易日开始重传
    // USTP_TERT_RESUME:从上次收到的续传
    // USTP_TERT_QUICK:只传送登录后公共流的内容
    pUserApi->SubscribePublicTopic(USTP_TERT_RESUME);
    // 设置飞马平台服务的地址，可以注册多个地址备用
    pUserApi->RegisterFront("tcp://172.28.21.133:15555");
    // 使客户端开始与后台服务建立连接
    pUserApi->Init();
    // 客户端等待报单操作完成
    WaitForSingleObject(g_hEvent, INFINITE);
    // 释放 API 实例
    pUserApi->Release();
    return 0;
}
```